Predicting Dementia: A Study Using Hand-Drawn Clock Images

By Stacey Beck and Ian Byrne

Coco Krumme, Ph.D., Eric Gilbert, Ph.D., Kevyn Collins-Thompson, Ph.D.

SIADS 694 & 695

Milestone II

University of Michigan Master of Applied Data Science

September 26th, 2021

Data Access

We used Google Colab Pro Plus to maximize RAM allowance as well as GPU compute. We have various .py files that contain functions to help automate processes such as extracting data, cleaning, labeling, creating dictionaries, and visualizing the images. We also use .py files for our models and import and run everything in notebooks to utilize the RAM and GPU that Google provides.

You can find all our work in our Repository on Github at: https://github.com/ian-byrne/MADSmilestone2

We recommend you open our presentation notebooks so you are not searching our repository for where to start.
CNN- Scores
CNN- Dementia Labels
Transfer Learning with K-Means

I. INTRODUCTION

Hospitals and clinical settings use the Clock Drawing Test (CDT) as a measure of cognitive function in regards to executive functioning like planning, numbering, as well as other areas such as concentration, and visual-spatial processing[1] to determine the presence of moderate to severe cognitive impairment, dementia, subtle cognitive impairment associated with Traumatic Brain Injury (TBI) and Alzheimer's Disease.[2,3] A clinician administers the test by providing the individual being tested with a blank piece of paper and asking the person to draw a clock with the hands pointing to "ten past 11 O'clock." This has generally been a successful tool.

Though the CDT has shown reliability as a screening tool for dementia, it has fallen short of being able to detect mild cognitive impairment or mild presence of dementia.[2,4] One idea in using the CDT to help catch or detect the development of any cognitive impairment or dementia before the disease has surfaced is through the use of machine learning for prediction. Our project is inspired by the AI Crowd's Clock Drawing Image challenge[5] to detect Alzheimer's Disease by using raw clock drawing images and health conditions through a series of ten different drawing rounds (the CDT is administered to each participant once per year over the course of ten years) in an effort to predict the presence of possible dementia, probable dementia or no cognitive impairment given a clock-drawing. This AI Crowd challenge partnered with the National Health and Aging Trends Study (NHATS) to procure the data. They also partnered with neuropsychologists to decompose the clock drawings into specific and important numerical features related to the diagnosis of dementia, such as position of clock hands, center dot, clock face, digits and perseveration.[5] These features were used for the challenge. We, however, did not have access to these valuable features for our machine learning project, so we had to approach it from the raw data.

II. INTENDED GOALS AND FINAL DIRECTION

The intended goals for this project were to better understand computer vision problems using images in a classification task working with Convolutional Neural Networks (CNN) and then test the system on generated images using Generative Adversarial Neural Networks. However, we came to realize that the data presented unforeseen challenges and this has led us to take on a new direction for the latter half of the project. The images we used for this project are unlabeled and the labeling systems known to us seem arbitrary and inconsistent. We chose to compare a couple of label systems for the images to use on our supervision task. One labeling idea was to look at whether or not the clock images themselves could alone indicate dementia, and another option was to see if the CNN could grade or score the images in a similar manner to the NHATs researchers. We achieved worse than random predictions using the scratch CNN architecture and decided to compare these results with a pre-trained ResNet50 model that yielded slightly better results, though still much room for improvement to classify the clock drawings. We suspect that the labeling system is one of our major bottlenecks among other issues such as sparsity of our image data, and therefore, wanted to experiment with clustering the images into the optimal number of classes using transfer learning to extract features in each clock drawing for our unsupervised task.

III. SUPERVISED LEARNING WITH A CNN

A. Motivation

We are interested in utilizing the images of clocks drawn by the subjects from NHATS to see if a CNN could predict whether or not a subject has dementia. We are both interested in implementing a CNN using Pytorch and leveraging GPU acceleration for this novel data set. Additionally, we want to see if a CNN can help remove the need for a person to score the drawings, which use a scale of 0 (worst) to 5 (best), and then pass those predictions along with other scores and measures from the NHATS testing to determine if a subject has dementia.

B. Data

We utilize two main collections of datasets[6] for our project from NHATS, one being Annual Rounds Data, which are ten separate Public Use STATA files containing detailed information about the participants, and the other being zipped Clock Drawing .TIFF files. There are ten separate STATA and zipped .TIFF files because the data was collected over the span of ten years. Special request to access these files is required and available as long as the user agrees to NHATS's Public Data use agreements. Each zipped .TIFF file contains between 3600 to 7000 (mostly)

[1] Eknoyan, D., Eknoyan, D., Hurley, R. A., Taber, K. H., (2012, July 1). The clock drawing task: Common errors and functional neuroanatomy. The Journal of Neuropsychiatry and Clinical Neurosciences. Retrieved August 3, 2021, from https://neuro.psychiatryonline.org/doi/10.1176/appi.neuropsych.12070180.
[2] Spenciere, B., Alves, H., & Charchat-Fichman, H. (2017). Scoring systems for the Clock Drawing Test: A historical review. *Dementia & Neuropsychologia*, *11*(1), 6–14. https://doi.org/10.1590/1980-57642016dn11-010003
[3] Hazan, E., Zhang, J., Brenkel, M., Shulman, K., & Feinstein, A. (2017). Getting clocked: screening for TBI-related cognitive impairment with the clock drawing test. *Brain Injury*, *31*(11), 1501–1506. https://doi.org/10.1080/02699052.2017.1376763
[4] Aprahamian, I., Martinelli, J. E., Neri, A. L., & Yassuda, M. S. (2009). The Clock Drawing Test A review of its accuracy in screening for dementia. *Dementia & Neuropsychologia*, *3*(2), 74–80. https://doi.org/10.1590/s1980-57642009dn30200002
[5] https://www.aicrowd.com/challenges/addi-alzheimers-detection-challenge
[6] https://nhats.org/researcher/data-access/public-use-files

grayscale images of size 3312 x 2560, for a total of over 51,000 images. Some images are formatted in RGB. The STATA data, once all ten rounds joined, contains over 62,000 records and over 1,000 columns. From the STATA files, we care about 13 of those columns for the final analysis and also create our own 'label' and 'rounds' columns.

a. Annual Rounds Data

Preprocessing the data first includes determining which round data columns are of interest. The images are not labeled so we came up with a couple of labeling strategies. We talked to Ankit Pandey, one of the support persons who run the AI Crowd Clocks challenge, to ask about how they derived labels for the images. Initially, we attempted creating labels using the AI Crowd method, which includes using the values contained in the variable *'hc1disescn9'* to help label the images. The variable *'hc1disescn9'* indicates whether someone has dementia/ Alzheimer's. A response of '1 YES' indicates that they have a confirmed diagnosis of Alzheimer's/ Dementia and '2 NO' indicates no diagnosis has been given. A value of '7' indicates that a response of '1 YES' has already been recorded in a previous round. AI Crowd labeled each image as either: 0 (Pre-Alzheimer's), 1 (Post-Alzheimer's), or 2 (Normal).[5] If a subject reports '1 YES' in the variable *'hc1disescn9'* in a subsequent year, all previous years receive the label 0 (Pre-Alzheimer's). All *'hc1disescn9'* reports of '1 YES' and '7' receive the label 1 - (Post-Alzheimer's) for the current and subsequent rounds. All the rest of the records receive the label 2 (Normal).

This label strategy seems arbitrary and especially so since we suspect many participants with dementia were able to sufficiently participate in this study even with their disease undetected by their physicians, especially in early dementia. The values contained in the variable *''hc1disescn9'* are also self-reported and are thus subjective where participants may falsely report their diagnosis or even mis-report their diagnosis. We sought to find a better labeling strategy and decided on creating labels around the NHATS's dementia classification decision rules.[7] We use these specific labels to identify if someone has 0 (Possible Dementia), 1 (Probable Dementia/(Likely Dementia) and 2 (No Dementia).

The NHATS's study looked at using cognitive tests to develop their classifications.[7] These cognitive tests fit into three domains, orientation, memory/recall, and executive functioning. For an orientation-type task, the subject is asked to answer questions such as "Who is the President of the United States right now?" and "Without looking at a calendar or watch, please tell me what day of the week it is."[8] A memory-type task involves recalling a list of ten words in a span of a couple of minutes. Measuring executive functioning corresponds to the clock drawing task. Based on the NHATS's study, the variables used to help classify '0' for "Possible Dementia" include one cognitive test score with a cut off <= 1.5 standard deviations below the mean. The variables used to help classify '1' for 'Probable Dementia' include *'hc1disescn9'* for diagnosis ('1 YES', '2 NO', '7') and if this is not provided we looked at *'cp1dad8dem'* which provides a diagnosing like score through the use of a proxy, two cognitive test scores with cut offs <= 1.5 SD below the mean. All records still left unlabeled receive a '2' for 'No Dementia." (See *Appendix I)* For further understanding of the specific variables used in the cognitive tests and used in this label strategy refer to *Appendix II*, which describes each domain.

We applied the NHATS's label strategy on the data and received mixed labels for some subjects throughout the span of their participation across the years. For some rounds, the subject is labeled as '0', other rounds the subject has a label of '2.' We toyed with the label strategy one step further by applying the AI Crowd's strategy with NHATS's to produce labels that were consistent from year to year. For example, we applied NHATS's label strategy first. If a subject presented with '0' at some point in the ten year study, the AI Crowd's label strategy would label all previous years as a '2' and all subsequent years as a '0.' We ultimately abandoned this hybrid approach and kept with the well-studied NHATS's label strategy, albeit its inconsistency when applied to the dataset from year to year for a given subject. For further understanding of how these label applications appeared for certain subjects, refer to *Appendix III*.

b. Image Data

Preprocessing image data includes standardizing the inputs, so all of our images were either converted to grayscale or RGB. The majority of the data originally was in grayscale, though some images were formatted as RGB. Our first approach was to convert to grayscale and used the Pillow library Image.convert() module. This process uses the ITU-R 601-2 luma transform, which is just multiplying the red, green and blue channels by a certain fraction to achieve black and white. Only when we introduced the ResNet-50 model did we need to convert all images to RGB, which was unforeseen at the beginning of the project. Standard inputs to a neural network are images at 224 x 224 or smaller and since each of our images are approximately 16.6" by 8.5" (3312 x 2560 pixels), we needed to convert to a size that the network could handle. We used Image.resize() default nearest-neighbor

[7] *NATIONAL HEALTH AND AGING TRENDS STUDY (NHATS), Classification of Persons by Dementia Status in the National Health and Aging Trends Study*. (2013, July). https://www.nhats.org/sites/default/files/inline-files/DementiaTechnicalPaperJuly_2_4_2013_10_23_15.pdf, page 2

[8] https://nhats.colectica.org/explore/concordance/example.org/bfbfbb0c-3ce2-40ab-9dfd-10e782209fa4/example.org/8b311a6d-cba0-467e-af99-69fb7604eb6e

interpolation and factorized the size to an input of 207 x 160 x 1. This lost a lot of data and so we settled on resizing at 368 x 284 x 1. (See *Fig.1 & Fig.2)*

Fig.1 Example of the images used initially in our analysis at 207 x 160 x 3 (converted to RGB from grayscale)

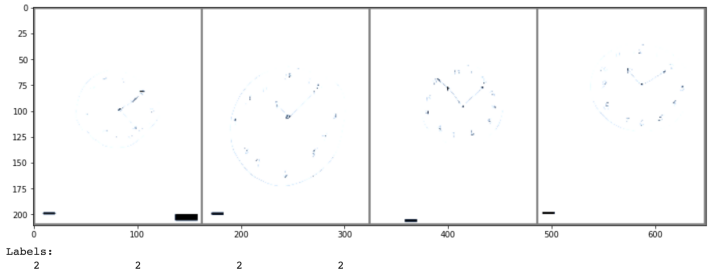
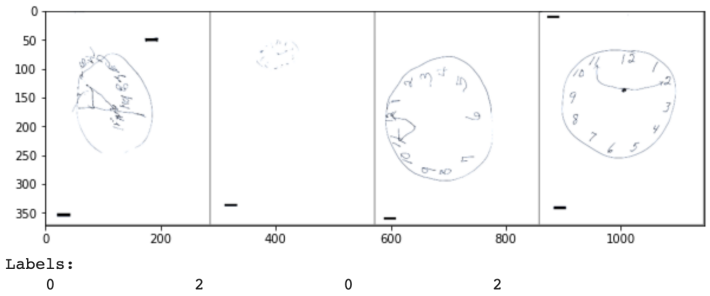


Fig.2 Example of the images used for our analysis at 368 x 284 x 3 (converted to RGB from grayscale)



c. Data Access

For ease of access, we loaded all our data, both the downloaded STATA and image files from our local directory and into Amazon's Simple Storage Service (S3) bucket. We accessed the data frame via the URL for the S3 object and loaded it with pandas. To access the images, we utilized the URL to understand the formats and get an idea of the processing that would need to be done and then created a DataLoader from Pytorch to loop through the images and download them to numpy files that contained the image arrays. We ran into issues with the set of data that was uploaded back to S3 in that while all of the images are of .TIFF data type, there are several types of images that can be stored as such and it took several days to sift through the proper protocol to read the images uniformly. Once we figured this out we created the numpy files and stored those arrays locally. We used the stored numpy files for the beginning phases of CNN training but soon realized the importance and need for converting images on the fly and eventually employed a data stream from S3 into the Pytorch data loader for training.

To access our images on S3 we created dictionaries of our dataframes where the keys are the rounds 1-10 and the data contained are arrays of tuples where the first tuple element is the subject's ID (*'spid'*) and the second tuple element is the label. Our data loader opens the image, converts to either grayscale or RGB and resizes the image. The image is then converted to a numpy array and is output as a tensor object before deploying to the CNN model for training. We converted the tensor to numpy on the held out test set for evaluation.

The data presented us with three more bottlenecks in the project. One issue is centered around organizing the data. In order to map the '*spid'* from the image data to the round data we contained the images in folders 1-10 on S3. We could not figure out how to stream through those folders to access all the data at once for training and therefore, had to break up training and testing into segments, which partitioned results, providing only insight to how certain rounds data performed versus the complete set of data. The second issue was the large imbalance of classes, where class 2 was predominant across all labeling strategies, but most pronounced for the AI Crowd label system. We organized our data in the dataframe and applied a random train, validation and test split. We balanced the train set by undersampling class 2 and randomly upsampled class 0 and 1 using image replacement rather than other sampling strategies such as SMOTE[9]. Re-balancing the train set is preferred over re-balancing the whole dataset since we want to assess the model's ability to predict our classes based on the true distribution of the test set. And lastly, we must consider that image resizing is losing a lot of data, which will impact the project from how

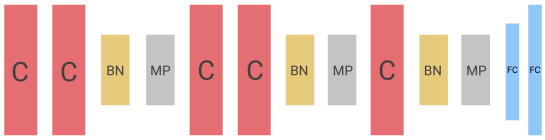
[9]Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953

much data the GPU can handle at a time to how well the model is able to learn based on important features still present in the image.

C. *Methods and Evaluation*

Our methods for this supervised task include building a CNN from scratch. We opted to use the Pytorch library and autograd engine to help build our network. We started by building a simple architecture inspired from the LeNet-5 architecture[10] and added to this base architecture to reduce overfitting and increase convergence of the loss curves between the training and validation sets. After much tuning, we settled on creating an architecture with 5 convolutions with ReLU activations, 3 batch normalizations, 3 max pooling, 2 drop outs and 2 linear layers. (See *Fig.3*) For more detail about network size see *Appendix III*.

Fig.3 Final CNN architecture



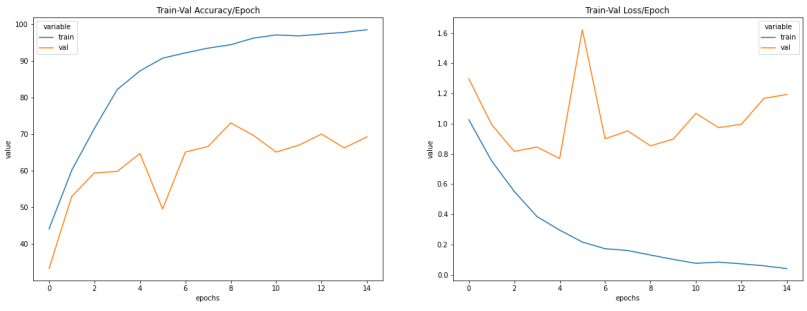
We optimized our model using stochastic gradient descent for the backward pass over the model and a softmax function applied to the final output layer during accuracy calculation. We use Cross Entropy to calculate our model loss as this has been shown to work well for multiclass classification models.[11] During our earlier training phases, we saw the validation loss plateauing and added a scheduler to help push the loss down after 2-4 epochs if the loss continued to increase in an effort to reduce overfitting. We added dropout layers as well as L2 regularization (See *Fig.4*) to the optimizer in the last model iteration to also help with model overfit and this helped produce the best results from our scratch CNN.

Fig.4 Cross Entropy applied with Softmax Activation Equation and L2 Regularization[11]

$$L(\hat{y}, y) = -\sum_{k}^{K} y^{(k)} \log \frac{e^{\hat{y}^{(k)}}}{\sum_{j=1}^{K} \hat{e}^{\hat{y}^{(j)}}} + \frac{1}{2}\lambda \|\mathbf{w}\|_2^2$$

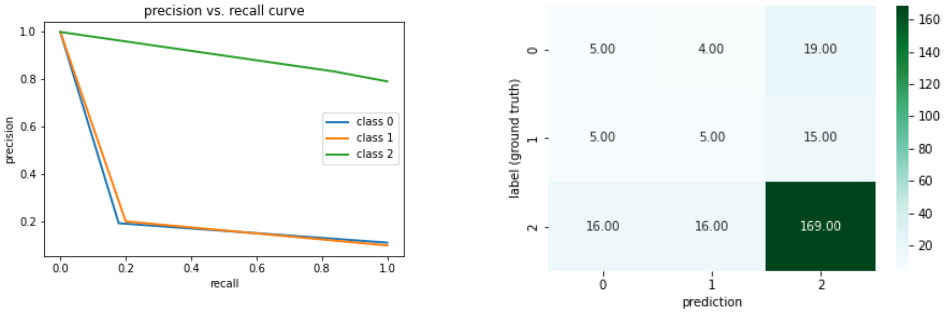
The general sequence of training went as follows: starting with training on round 1, round 5, round 6, round 7, round 9, round 2, round 8, round 4, round 3, and lastly round 10. There are missing data in round 10 and we spent some time understanding how to work around this issue in order to complete the training over all our data. Our best model output was model 4 that contained training on rounds 1, 5, 6 and 7 (over 11,000 training images). After training the model sequentially, we believe our model continued to overfit the training data. (See *Fig.5*) Though our training accuracy was high and best model validation and test accuracies were 69% and 70%, respectively, we saw low f1 scores and poor precision and recall scores. (See *Fig.6*) Our model predicted slightly better than random. (See *Appendix V for ROC and Classification Report*)

Fig.5 Training and Validation Accuracy and Loss for Model 4 of Scratch CNN



[10] Alake, R. (2020, June 25). *Understanding and Implementing LeNet-5 CNN Architecture (Deep Learning)*. Medium. https://towardsdatascience.com/understanding-and-implementing-lenet-5-cnn-architecture-deep-learning-a2d531ebc342
[11] Zafar, I. (2018, August). *Hands-On Convolutional Neural Networks with TensorFlow*. O'Reilly Online Learning. https://www.oreilly.com/library/view/hands-on-convolutional-neural/9781789130331/7f34b72e-f571-49d2-a37a-4ed6f8011c93.xhtml

Fig.6 Precision and Recall on Rounds 1, 5, 6, and 7 of Scratch CNN



We trained the ResNet-50 architecture, pre-trained on ImageNet, on our data using the NHATS label strategy for rounds 1, 5, 6, 7, 10, 2 and 3 but kept the best model, model 3 which was trained on rounds 1, 5 and 6. Our results improved slightly compared to the scratch CNN when looking at the precision and recall scores since accuracy was just measuring our overfitting to the data, particularly label 2. (See *Fig.7 & Fig.8 & APPENDIX V*)

Fig.7 Training and Validation Accuracy and Loss for Model 3 of re-trained, pre-trained ResNet-50

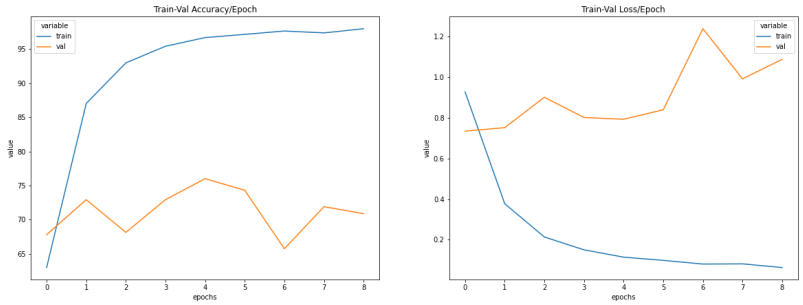
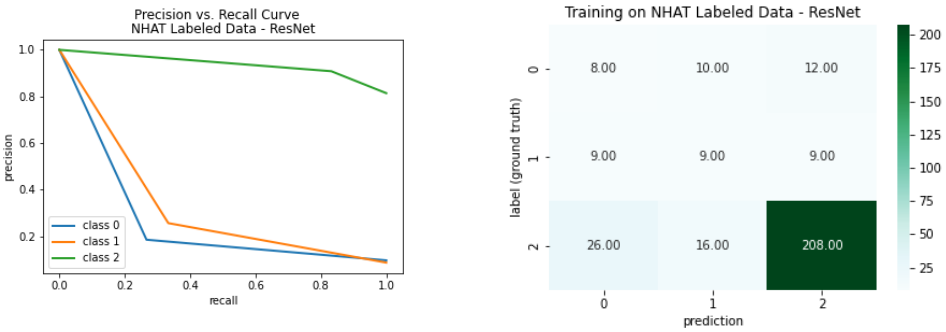


Fig.8 Precision and Recall on Rounds 1, 5,and 6 of re-trained, pre-trained ResNet-50



Even though we balanced our training set before validation and testing, the CNN and ResNet models continued to overfit to label 2. We might be seeing one class predicted more than others since most of the images are generally the same looking; circular objects, drawn to a particular size on the general area of the paper, creating a bias to the data. Additionally, because the images are so sparse with only a small percentage of pixels containing relevant information, the models may be classifying the images based on irrelevant features like the black boxes, noise in the sparsity, or even the global clock drawing placement on the page instead of the waviness of the lines or positioning of the digits and clock hands. Since most of the clock drawing images regardless of true class have these consistent global features, the model will tend to predict the most seen class in the test set.

We really care most about the recall score for this particular task since we would not want to predict someone with dementia as not having dementia since this could lead to unintended consequences. We are not overly predicting someone as having no dementia when they possibly could or do have dementia.

Similarly, we trained the ResNet-50 model on the score data using all rounds. This improved the grading of the images compared to the from scratch CNN, but ended up having a weighted average F1 score of .57. (See *Fig.9*) This is interesting as all of the data used to score the image is present within the image compared to the end diagnosis which, while using the images, also uses the other NHATS tests. If more accurate, this prediction method could remove the manual step of grading the images and have a single standard to grading as opposed to the subjective nature of having different graders for different patients. (See *Fig.10 for model accuracy*)

Fig.9 Some classes appeared to score better than others. In one round, scores 4 and 5 performed better as evidenced by the Area Under the Precision Recall Curve. In another round, class 0 was not being predicted, yet class 3 and 4 were highly predicted.

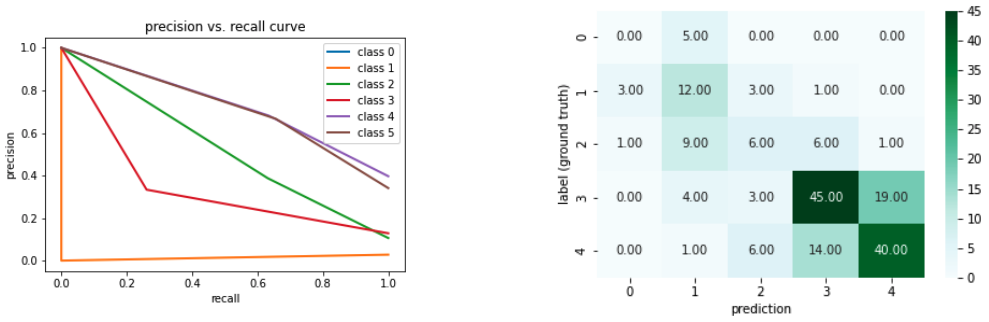
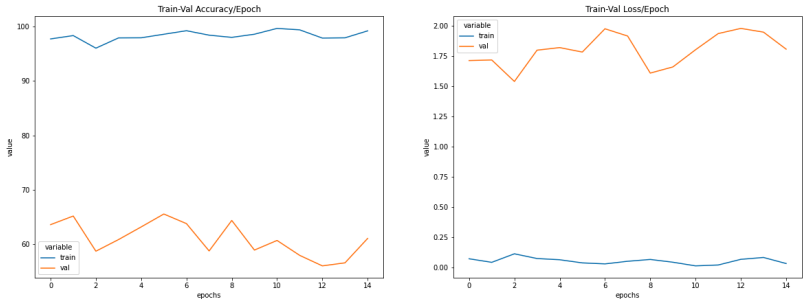


Fig.10 ResNet-50 model score data. Training data not learning, showing plateau as evidenced by Loss and Accuracy Curves



D. Other Analyses

We briefly looked at the full dataset from NHATs which utilized the assigned clock grades for the images as well as the other tests administered to see how classical supervised methods would work on that dataset. Part of this was to see if once a model was generated that could accurately score the images on the 0 to 5 scale, those scores could then be combined with the rest of the NHATs tests and utilized to make dementia classifications that way. This was run with the *'hc1disescn9'* variable removed to avoid any potential data leaks. These models scored very well, with the best model being that K nearest neighbors using the 3 neighbors. Another test was run to see how sensitive the data was to the image scores by removing them and seeing how that affected the overall accuracy of the models. Removing the image scores did negatively affect the accuracy, generally resulting in a drop of 2 percent for both the K nearest neighbor and Random Forest models.

We also ran PCA on this data to see the variance explanation over the principal components. Over 95% of the variance seen within the data frame with all but the one test can be explained by the first four principal components. This could potentially be a step utilized after the image scores are predicted in order to keep the number of features to a small number.

E. Failure Analysis

One area of failure was the inability of the CNN to correctly classify clock images. Regardless of the amount of tweaking made to the model, the highest f1 score we could achieve in that task was .33 when attempting to grade the images on a 0 through 5 scale. There are several theories as to why this played out as such. One of which is the size of the images. The original scanned drawings of the clocks were all the size of a standard sheet of paper which measures 3312 by 2560 pixels. This posed a considerable problem since a more standardized input image would be between 128 x128 and 256 x 256 pixels.[12] We originally ran the model with inputs of 207 by 160 pixels. This number was arrived at by taking the largest common factor of the original dimensions and dividing by the factor. In doing this, a significant amount of the images were degraded. This process was agnostic to the size of the original clock drawing and what section of the page it was drawn on. This meant a drawing that was made in the top left quadrant of the sheet lost more data than a drawing that was centered and took up a majority of the sheet. A potential remedy to this would be to use object detection to isolate the clock image before resizing, this idea is expanded on in the discussion section.

Another issue that arises in hand labeled image data such as this is that scoring is subjective and criteria can be interpreted differently by individual graders. Due to the NHATs instructions that a square or round clock is an acceptable drawing, there is potential to throw off an automated grading system, biasing to one shape or the other regardless of the quality of the drawing. Utilizing a more exact image of the clock should be able to help mitigate this.

Lastly, we are faced with much bias in the way of the labeling systems available to us as well as how the image data was preprocessed and utilized for training. Had we been able to capture just the relevant pieces of the clock images rather than the sparsity of the page along with other artifacts, such as the black boxes, we would have been able to eliminate some of the bias we were seeing in our results.

IV. UNSUPERVISED TRANSFER LEARNING WITH PCA AND K-MEANS FOR CLUSTERING

A. Motivation

Our motivation with this unsupervised learning task stems from our skepticism around the labeling strategies of the image data. We aim to better understand which is the optimal number of classes for this data and how the images cluster. We decided to use a transfer learning technique by fitting some of our data to the output features from our CNN and the pre-trained ResNet-50 models in order to extract the feature representation of our images and reduce these large features sets using Principal Component Analysis (PCA) to feed into a K-Means model for clustering.[13]

B. Data

We used the same set of procedures for our dataset as we employed in our supervision task such as splitting and balancing the data and accessing from S3. We only ran our models arbitrarily with round 7 data as to save compute time and to get a sense of how K-means would benefit us as a model choice in our analysis. Data transformation details are included in the methods section.

C. Methods and Evaluation

In order to determine the optimal number of clusters for our data we applied a transfer learning technique utilizing the features from our learned models, both the pre-trained ResNet-50 on ImageNet model and our scratch CNN. The ResNet-50 model outputs 2,048 features and our CNN outputs 640,000 features before the final connected layers of the network, which reduces these features to a specific number of classes. For both models, the final fully connected layers (class layers) were removed prior to fitting. The data was read in as tensors, then changed to numpy arrays and the dimensions flattened to the length of the array and combined feature values. Using scikit-learn's decomposition package PCA module, we reduced the features using a variance metric of .98 to ensure a more accurate model. The output features after PCA for the CNN reduced from 640,000 to 221 and the ResNet-50 model from 2,048 features to 114. Lowering the variance would reduce the number of final features at the cost of accuracy.

We then fit the K-means model using scikit-learn's KMeans() module to extract the number of clusters as well as the images' corresponding cluster labels from the images feature outputs from the CNN and ResNet-50 models. We can see from obtaining the silhouette scores and the mean squares that the ideal number of clusters for

[12] Rukundo, O. (2020). *EFFECTS OF IMAGE SIZE ON DEEP LEARNING*. https://arxiv.org/pdf/2101.11508.pdf
[13] Cohn, R., & Holm, E. (2021). Unsupervised Machine Learning Via Transfer Learning and k-Means Clustering to Classify Materials Image Data. *Integrating Materials and Manufacturing Innovation*, *10*(2), 231–244. https://doi.org/10.1007/s40192-021-00205-8

round 7 data is either 2 or 6 clusters using the data from our CNN. The clusters resulting from the image feature outputs from the ResNet model are either 2 or 3. (See *Fig.11 & Fig.12*)

Fig.11 Scratch CNN trained with half the dataset, fit to test set from round 7

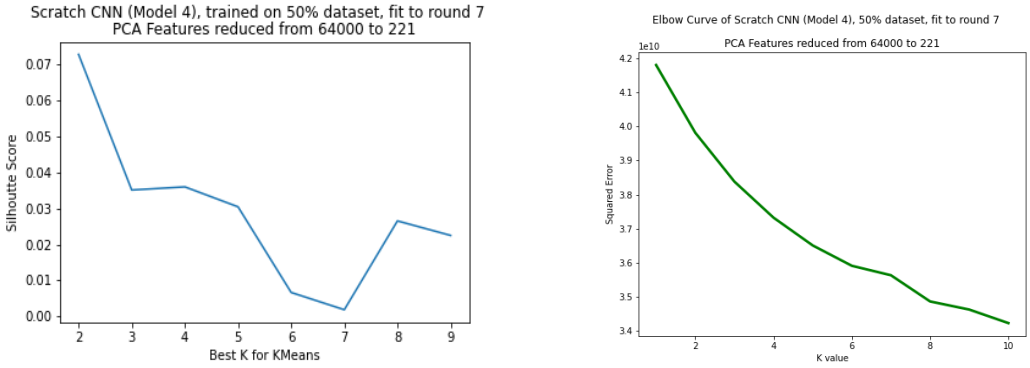
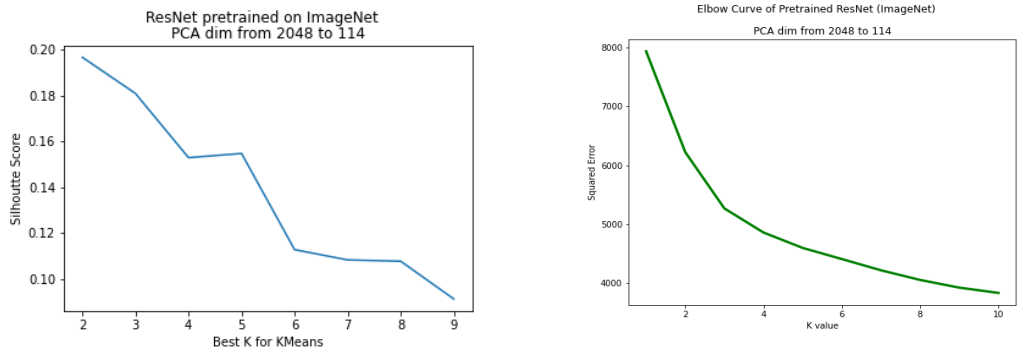


Fig.12 Pretrained ResNet-50 on ImageNet, fit to test set from round 7



We came back to this problem full circle and ran the K-Means model on the image feature outputs using 3 clusters and visualized the clusters. Unless the images were in grayscale, the clusters appeared noisy. (See *Fig.13*)

Fig.13 Grayscale 207 x 160 x 1 images example of clusters

K-Means predicts poorly against the target labels once accounting for the permutation of the K-Means cluster labels. We found that the model clustered our test set of images as either class 2 or class 1, depending on the balance of classes in the dataset. There is likely nothing meaningful to gather from these results and the results may be related to how sparse and globally similar these data are. There are consistent features in each image across all images, one being a black box at the bottom left corner of the paper hiding sensitive patient information. The fact that we did not remove this black box appears to be impacting the analysis of our unsupervised approach, as evidenced by the cluster image in figure 13.

V. DISCUSSION

We were very surprised at the inability of the neural network to reliably learn either the final target labels or the image scores from the drawings of the clocks. On the surface, the concept of being able to score an image of a clock does not seem to be very difficult, but once the image resizing and presence of the black boxes are taken into account, it clearly becomes much more difficult. The size of the images posed significant challenges as compressing them to a reasonable size for analysis seemed to exacerbate the labeling issues by losing data. Many of the improvements that could be made involve different ways to resize the images to lose less data as well as remove any potential noise.

One option that could allow for more accurate labeling of the clock images whether to determine dementia or scoring the drawing would be to identify the bounds of the drawn clock and focus in on that area of the full image and crop before the clock is resized. To do so we could use a form of object detection. This would allow us to maximize the amount of data captured regarding the clock images, reducing sparsity and noise, as well as potentially avoid the black boxes that seem to be present in the majority of images.

A simple solution to this would be to determine the top, bottom, left and right pixels of the image using numpy, build in a layer of padding, and then crop the square or rectangle identified as the image. Using this method would require some way to avoid the black boxes since a naive algorithm will use the pixels present in them to determine bounding box coordinates which may dramatically impact the effectiveness of the resizing. This would also still keep the box in the image to be analyzed which we believe is one of the main reasons the network has a hard time learning.

Using a more advanced machine learning object detection API would potentially prove far more useful. YOLO v3[14] or the object_detector_app[15] could be options to use. These models would allow for identification of both the black boxes and the block images separately which would allow us to focus on the clock image alone while resizing. This may also assist in understanding the features that the network is using to label images by allowing us to potentially identify circle vs. square and determine which numbers are indicated on the clock face, or how many hands appear on the clock. A downside to a process such as this is that it has the potential to cause training to take much longer than it currently does. For reference, training ResNet-50 on all 10 rounds to predict clock scores took roughly 8 hours total and needed to be attended to in between rounds with our current dataset definition.

Another extension of the project could include using a CNN to look at the time series nature of the rounds.[16] In our current analysis we only looked at each round individually. It may be possible for us to devise a CNN that can identify the changes between rounds and determine dementia labels from this. If reasonable labeling is possible with this approach it would be especially interesting to compare the round at which the CNN predicted the patient had dementia compared to when, if at all, the patient was diagnosed by a medical doctor. A model that uses this approach seems relatively straight forward using the NHATS's dataframe data containing all of their tests, but more complicated if we are to use just the images and compare them over time. We need to conduct more research to see if this approach is viable.

We could also attempt fully representing the image as a vector of the image coordinates, removing the white space and feeding through a neural network for training. Some caveats with this solution are that not all images are of equal size, some clocks are drawn to fill the page while others can fit in the page's corner. Models will require equal size inputs. Also, we would be capturing the black box with this approach, and would need to address this issue before creating the vector representation of just the clock image.

One of our main takeaways from this project has been the importance of the quality of our image data. While we know this seems obvious, having experienced significant setbacks due to image data types, image size, or noise within the image, have solidified the importance of having a solid grasp of the data to be used in our models. As mentioned above, many of the improvements that can be made to the models and the project involve more accurately identifying the clock drawing in order to lose less information about it once it is resized.

We were hopeful that our unsupervised method of using PCA with K-Means to cluster our images would provide a bit more insight into the label system utilized throughout the project. However, we come to realize that the analyses do not provide any real conclusion to our questions. This is because these models are calculating distance metrics on sparse data that likely result in meaningless values. The data are not dissimilar enough for the model to

[14] Redmon, J. (2020). *YOLO: Real-Time Object Detection*. YOLO: Real-Time Object Detection. https://pjreddie.com/darknet/yolo/
[15] D. (2017). *GitHub - datitran/object_detector_app: Real-Time Object Recognition App with Tensorflow and OpenCV*. GitHub. https://github.com/datitran/object_detector_app
[16] Fawaz, I. H. (2019, March 2). *Deep learning for time series classification: a. . .* Data Mining and Knowledge Discovery. https://link.springer.com/article/10.1007/s10618-019-00619-1?error=cookies_not_supported&code=6614f4cd-1840-41a5-a038-1337422fabd3

detect and had a hard time capturing the differences that we were interested in, like the details contained without the clock drawing image alone. Our images were originally represented as binary 0's and 1's and have been converted in the process to RGB but the same data is contained within this conversion. When the image features were reduced using variance of .98, it kept more feature components than were likely helpful to represent the data more accurately. These images contain many artifacts and unrelated objects other than the clock itself (black boxes) that are consistent from image to image that these components could have been considered over the actual clock drawing itself, especially smaller clock images, which appeared to be drowned out after feature reduction and clustering. After clustering, what resulted was just noise, unless the data was kept in grayscale, but then those images with smaller or lightly drawn clocks were reduced and the black box at the corner of the page was represented. Utilizing a different architecture called Sparse K-Means[17] for clustering could be a next step after applying improvements to the final preprocessing state of each image.

A. Ethical Issues

Caution and care are required if we were to build a robust model to predict whether someone may possibly have dementia, likely has dementia or does not have dementia given a clock drawing. Precision and recall scores poorly predicted the less represented classes, 0 and 1. The sensitivity in predictions is important especially if we want to capture the disease rather than miss it. Recall scores would need to be high in this case where we do not really care about false positive cases (predicting disease when there is none). False negative cases, even given training half the image dataset on the Resnet-50 model, were obtaining unsatisfactory results.

Considerable biases exist within the image itself as well as the labeling systems used. There is too much sparsity contained in each image as well as black boxes, which appear to be affecting model learning. The labels are created from partially self-reported data that is also inconsistent for a given subject through the course of the study. These biases are represented in our results, which hurt the ability to predict a class given a drawing.

When using machine learning in healthcare it is important to be able to explain the model to the patient or doctor in clear terms. This should be taken into account when building the model and selecting features. If we utilized an object detection algorithm to detect not only the clock but also numbers and hands on the clock this would make the explanation of features easier to understand, especially if the patient, their family, or doctor is not familiar with the process of machine learning. The clock drawing algorithm should be used as an adjunct to professional experience in making diagnoses.

VI. STATEMENT OF WORK

We each applied ourselves in every aspect of this project. We worked together to store the data in S3. Stacey was in charge of cleaning and labeling the data as well as creating the dictionaries and numpy files. Ian was responsible for creating the Resized Clock data loader class, which helped with streaming images. Stacey and Ian both tackled improvements in the functions and classes created to run image preprocessing and getting and streaming data. Stacey was responsible for building the CNN, metrics and visualizations as well as transfer learning and K-means. Ian was responsible for running the scores data through the CNN as well as other multimodel tests such as K-NN and Randomforest for the rounds data. We both were involved in the write-up and preparation of the presentation notebooks.

[17] Witten, D. M., & Tibshirani, R. (2010). A Framework for Feature Selection in Clustering. *Journal of the American Statistical Association*, *105*(490), 713–726. https://doi.org/10.1198/jasa.2010.tm09415

APPENDIX I

Table 2. Criteria for dementia classification and unweighted Ns by age group

Dementia classification	Probable dementia			Possible dementia	No dementia
Criteria	Diagnosis reported	Met AD8 criteria if no diagnosis reported (proxy only)	≤ 1.5 SDs below mean in at least 2 domains	≤ 1.5 SD below mean in 1 domain	All others
Persons 65+ [1]	457	159	422	996	5,575
Person 71+ [1]	435	140	393	878	4043

[1] Total N for 65+ = 7609; Total N for 71+ = 5889. Excludes nursing home residents: 468 persons 65+ and 458 persons 71+. See Table 5 for dementia classification of nursing home residents.

Score cutpoints developed using weighted data are shown in Table 3. Self-respondents who refused a test or answered don't know or were unable to do a test were scored as 0.

Table 3. Score cutpoints for ≤ 1.5 SDs below mean on NHATS cognitive domains

Domain	Orientation	Memory	Executive functioning
Score range	0 to 8	0 to 20	0 to 5
Score cutpoints	≤ 3	≤ 3	≤ 1

Source: *NATIONAL HEALTH AND AGING TRENDS STUDY (NHATS), Classification of Persons by Dementia Status in the National Health and Aging Trends Study*. (2013, July). https://www.nhats.org/sites/default/files/inline-files/DementiaTechnicalPaperJuly_2_4_2013_10_23_15.pdf, page 4

APPENDIX II

- *Orientation*
    - President and Vice President First and Last names: *'cg1presidna1', 'cg1presidna3', 'cg1vpname1', 'cg1vpname3'*
    - Date, Month, Year, Day of the Week: *'cg1todaydat1'* (Month), *'cg1todaydat2'* (Day), *'cg1todaydat3'* (Year), *'cg1todaydat4'* (Day of the Week)
    - Each correct answer gets a point; Total points out of 8.
    - Score cut point for <= 1.5 SD below the mean is <= 3 points across all variables
- *Memory*
    - Delayed Word Recall: '*cg1dwrdimmrc*' (total Score of some number out of 10 points)
    - Immediate Word Recall: *'cg1dwrddlyrc'* (total Score of some number out if 10 points)
    - Total points out of 20
    - Score cut off is <= 3 points across all variables
- *Executive Functioning*
    - Clock Drawing Battery: *'cg1dclkdraw'*
    - Total points out of 5
    - Score cut off is <= 1 point

APPENDIX III

NHATS Label System Applied

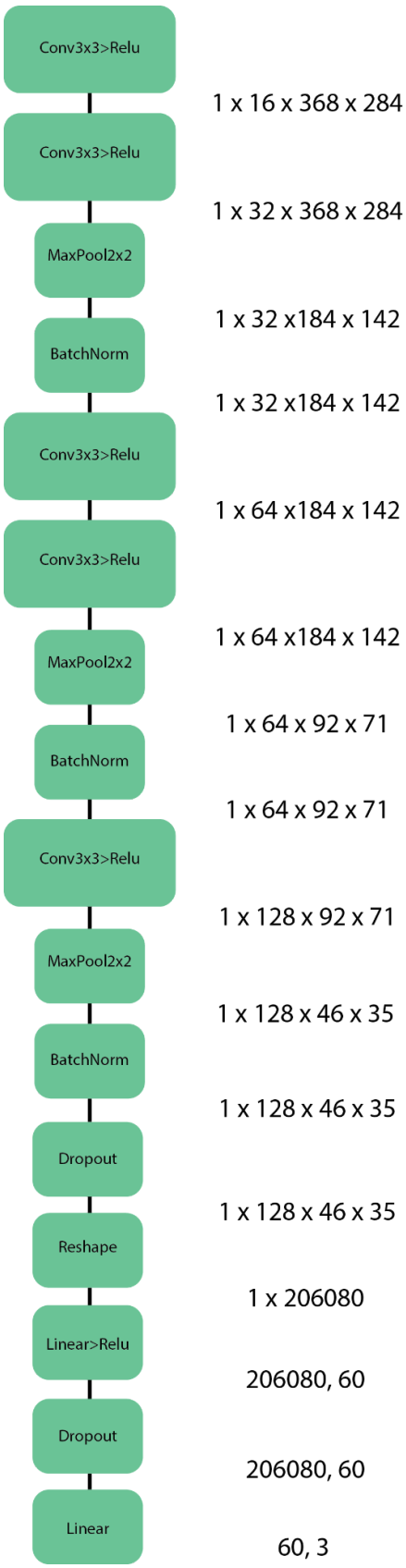
	round	spid	label
17	1	10000024	2
8258	2	10000024	0
16510	3	10000024	0
22100	4	10000024	1
	round	spid	label
34111	5	20006986	1
41398	6	20006986	2
47726	7	20006986	2
53278	8	20006986	0
58262	9	20006986	1
62655	10	20006986	0
	round	spid	label
28	1	10000041	0
8268	2	10000041	2
16744	3	10000041	2
22301	4	10000041	2
25869	5	10000041	0
34203	6	10000041	2
41477	7	10000041	0
47788	8	10000041	0
53333	9	10000041	2
58309	10	10000041	2
	round	spid	label
8321	2	10000131	2
18110	3	10000131	2
23412	4	10000131	2
25902	5	10000131	0
34233	6	10000131	2
41504	7	10000131	2
	round	spid	label
120	1	10000175	2
8348	2	10000175	2
19596	3	10000175	2
24611	4	10000175	2
25917	5	10000175	2
34246	6	10000175	0
41517	7	10000175	0
	round	spid	label
34	1	10000047	2
8274	2	10000047	1
20258	3	10000047	1
25149	4	10000047	1
25872	5	10000047	1
34206	6	10000047	1
41479	7	10000047	1
53335	9	10000047	1
	round	spid	label
34152	5	20007062	1
41432	6	20007062	0
47753	7	20007062	0
53302	8	20007062	1
58281	9	20007062	1
62673	10	20007062	1

Hybrid Label System Applied

	round	spid	label
17	1	10000024	0
8258	2	10000024	0
16510	3	10000024	0
22100	4	10000024	1
	round	spid	label
34111	5	20006986	1
41398	6	20006986	1
47726	7	20006986	1
53278	8	20006986	1
58262	9	20006986	1
62655	10	20006986	1
	round	spid	label
28	1	10000041	0
8268	2	10000041	0
16744	3	10000041	0
22301	4	10000041	0
25869	5	10000041	0
34203	6	10000041	0
41477	7	10000041	0
47788	8	10000041	0
53333	9	10000041	0
58309	10	10000041	0
	round	spid	label
8321	2	10000131	2
18110	3	10000131	2
23412	4	10000131	2
25902	5	10000131	0
34233	6	10000131	0
41504	7	10000131	0
	round	spid	label
120	1	10000175	2
8348	2	10000175	2
19596	3	10000175	2
24611	4	10000175	2
25917	5	10000175	2
34246	6	10000175	0
41517	7	10000175	0
	round	spid	label
34	1	10000047	0
8274	2	10000047	1
20258	3	10000047	1
25149	4	10000047	1
25872	5	10000047	1
34206	6	10000047	1
41479	7	10000047	1
53335	9	10000047	1
	round	spid	label
34152	5	20007062	1
41432	6	20007062	1
47753	7	20007062	1
53302	8	20007062	1
58281	9	20007062	1
62673	10	20007062	1

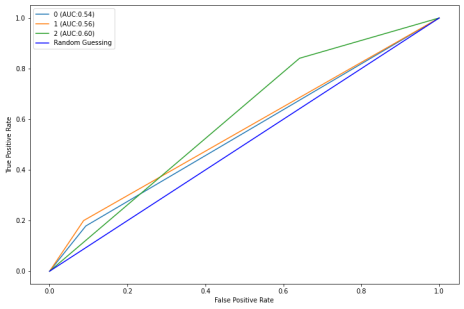
APPENDIX IV

Final Convolutional Neural Network before Training on ResNet

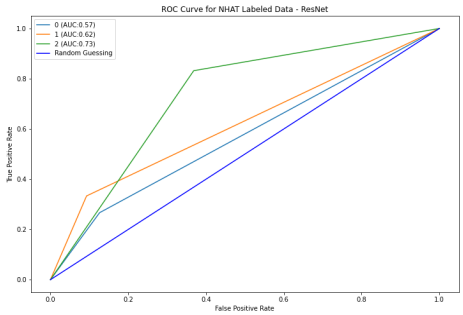


APPENDIX V

ROC Curve on CNN for best model using NHATS Labels



ROC Curve on Pre-trained ReNet50 for best model



Classification Report of CNN best model

	Possible Dementia (0)	Likely Dementia (1)	No Dementia (2)	accuracy	macro avg	weighted avg
precision	0.192308	0.2	0.832512	0.704724	0.408273	0.699683
recall	0.178571	0.2	0.840796	0.704724	0.406456	0.704724
f1-score	0.185185	0.2	0.836634	0.704724	0.407273	0.702160
support	28.000000	25.0	201.000000	0.704724	254.000000	254.000000

Classification Report of Pre-trained ResNet-50 of best model

	Possible Dementia (0)	Likely Dementia (1)	No Dementia (2)	accuracy	macro avg	weighted avg
precision	0.186047	0.257143	0.908297	0.732899	0.450495	0.780451
recall	0.266667	0.333333	0.832000	0.732899	0.477333	0.732899
f1-score	0.219178	0.290323	0.868476	0.732899	0.459326	0.754179
support	30.000000	27.000000	250.000000	0.732899	307.000000	307.000000