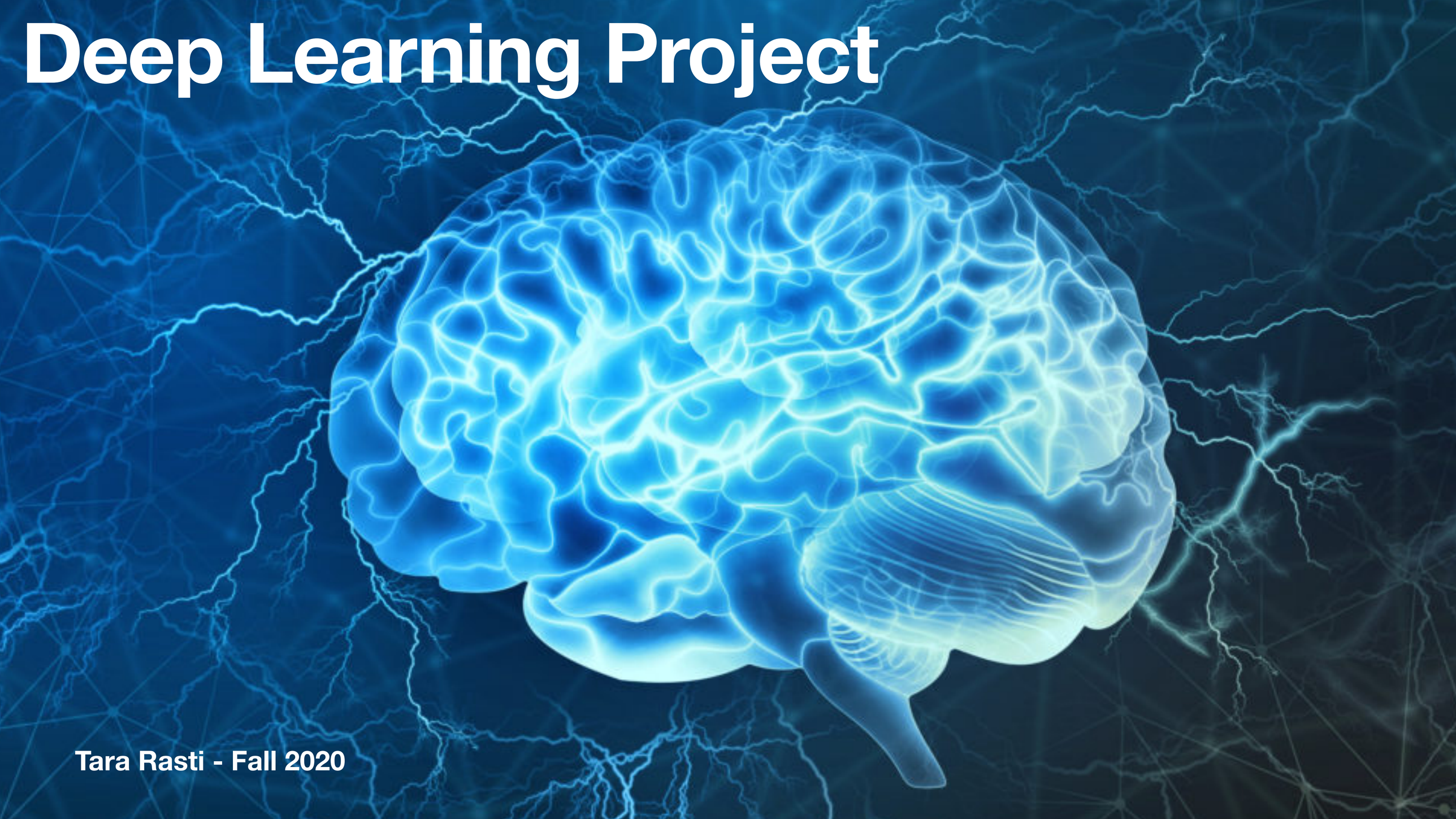
Deep Learning Project



Tara Rasti - Fall 2020

Alzheimer’s Disease Diagnostic by Adaption of Convolutional Network

What is Alzheimer Disease?

- a progressive brain disorder
- the most common case of dementia
- AD leads to the death of nerve cells and tissue loss throughout the brain, thus reducing the brain volume in size dramatically through time and affecting most of its functions.
- one out of 85 persons will have the AD by 2050
- sixth-leading cause of death in the United States

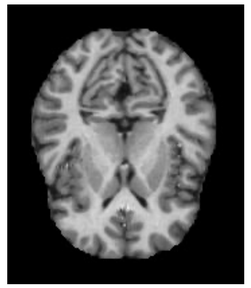
Motivation

- Early diagnosis, playing an important role in preventing progress and treating the Alzheimer’s disease
- the necessity of having a computer-aided system for early and accurate AD diagnosis becomes critical
- Several popular non-invasive neuro-imaging tools : sMRI, fMRI, PET
- sMRI has been recognized as a promising indicator of the AD progression
- Convolutional neural network, which has shown remarkable performance in the field of image recognition, has also been used for the diagnostic classification of AD with multimodal neuro-imaging data

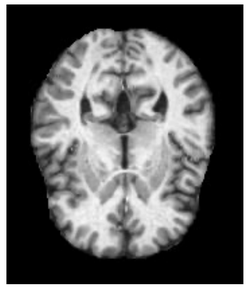
Dataset

https://www.kaggle.com/legendahmed/alzheimermridataset

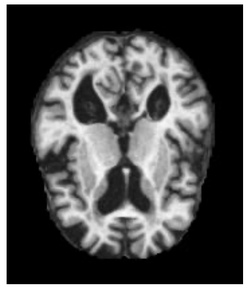
- consists of 6,400 MRI axial slices
- is categorized into four groups : non demented, very mild demented, mild demented and moderate demented
- The dimensions of all photos are 176 * 208 and all of them are jpg



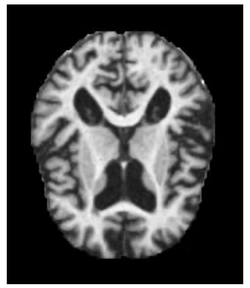
NonDemented



VeryMildDemented



MildDemented



ModerateDemented

The Model

CNN Models

I applied three CNN models :

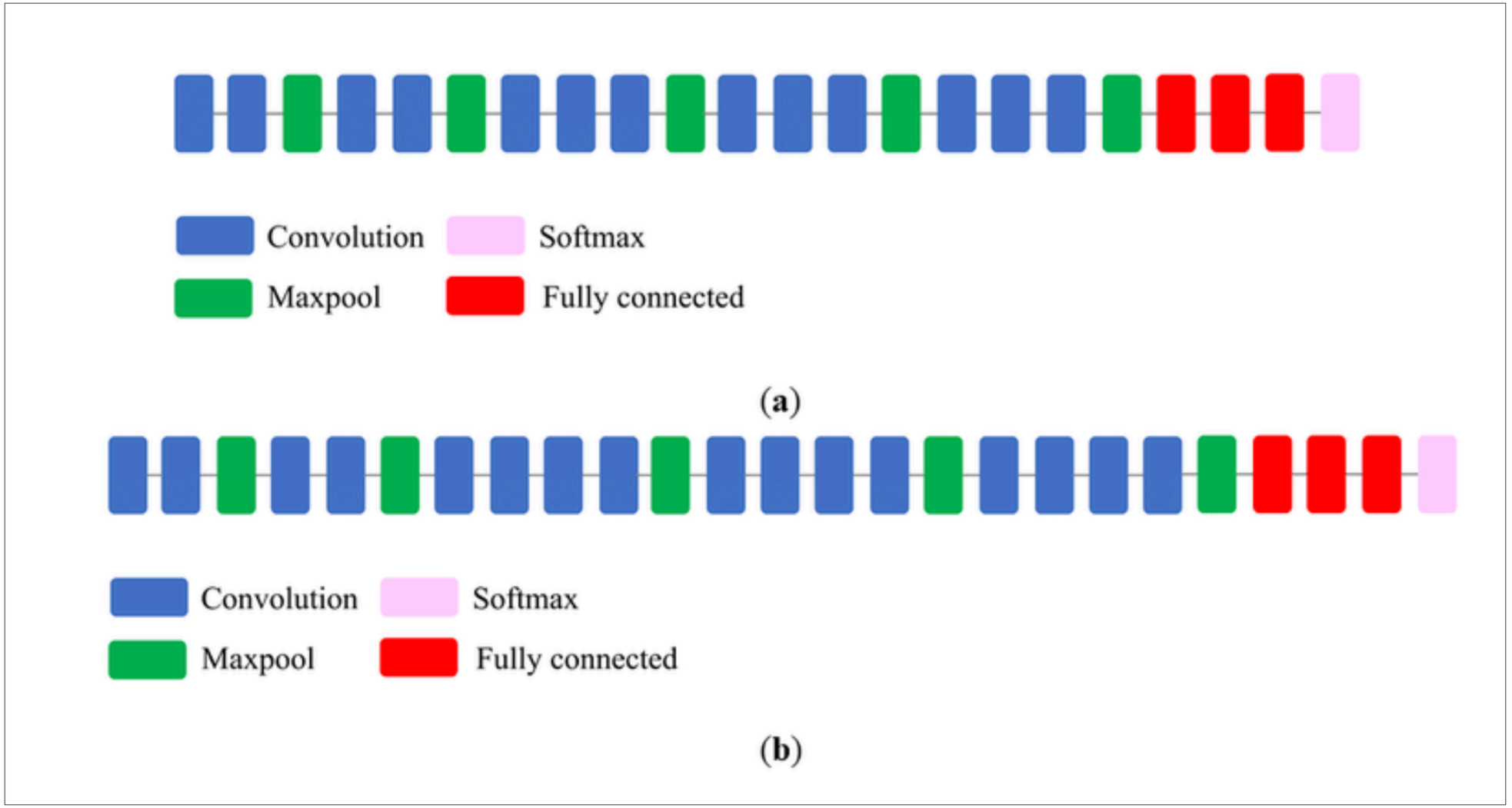
- My Model : a CNN model with

  5 convolutional layers,

  3 max pooling layers and 3 linear layers

Conv+ReLU
MaxPool
Conv+ReLU
MaxPool
Conv+ReLU
MaxPool
Conv+ReLU
Conv+ReLU
Fully Connected+LeakyReLU
Fully Connected+LeakyReLU
Fully Connected+Softmax

- VGG16 : VGG-16 is a convolutional neural network that is 16 layers deep
- VGG19 : VGG-19 is a convolutional neural network that is 19 layers deep



Schematic Diagram of (a) VGG16 and (b) VGG19 models

Model Details

Regularization Techniques Used

- Dropout : 0.3
- Weight Decay : 1e-5
- Data Augmentation : Random Rotation - Horizontal Flip

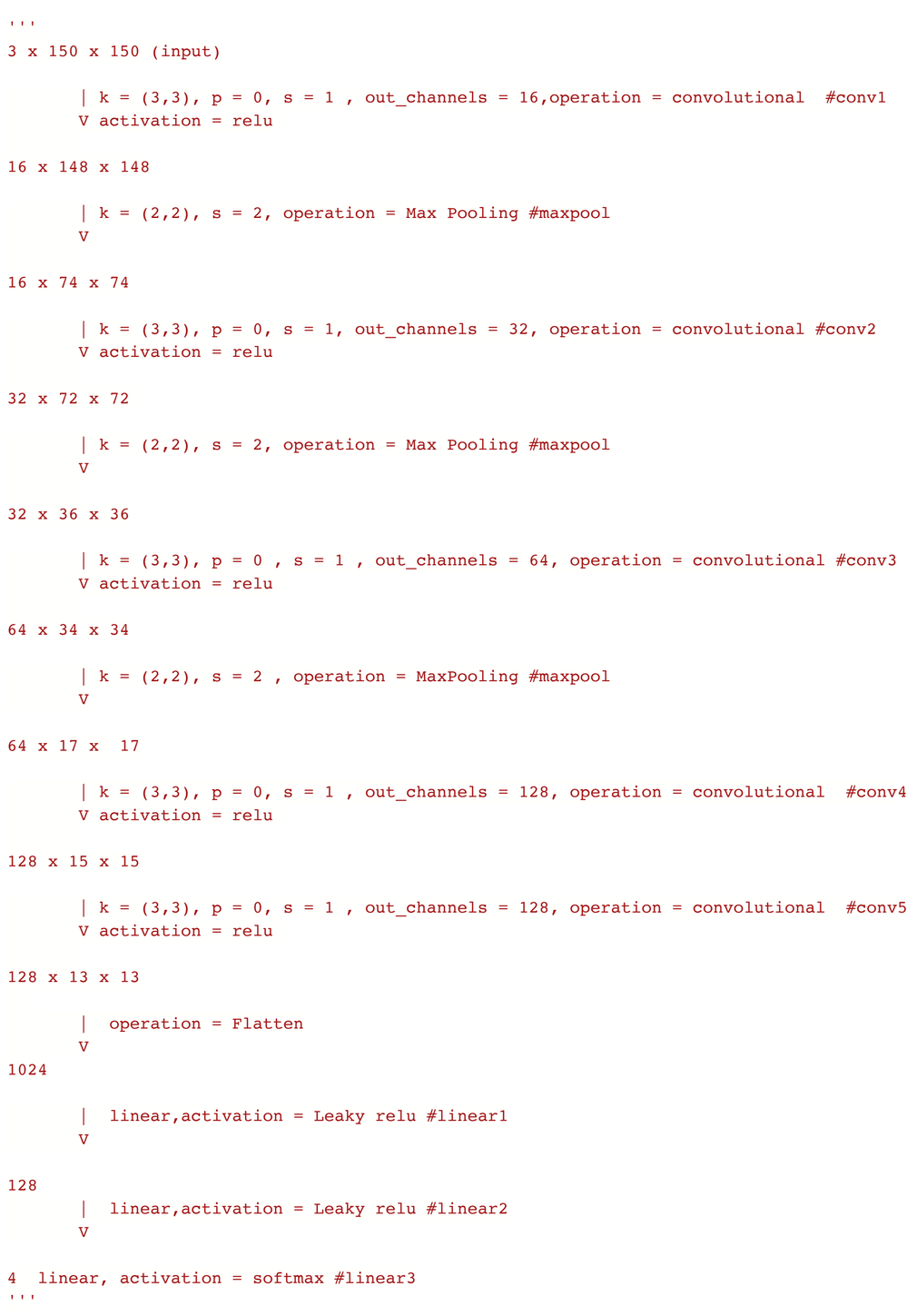
Loss Function

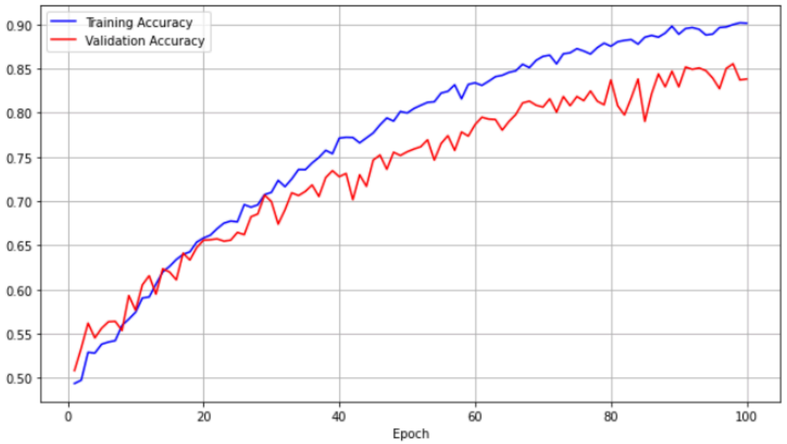
Cross Entropy

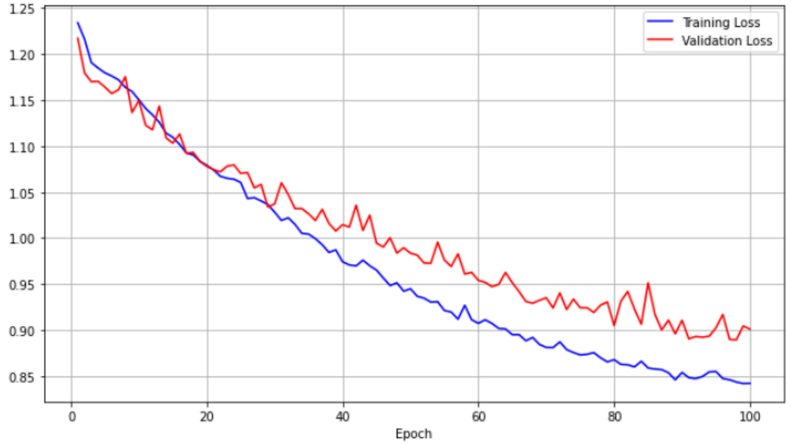
Table Showing Accuracy for all Implemented Models

Trial	Model	Validation Ratio	Optimizer	Epochs	Accuracy(val)	Accuracy(train)
1st	CNN	0.2	Adam	100	65%	87%
2nd	CNN	0.2	Adam	200	68%	96%
3rd	VGG16	0.2	SGD	20	75%	95%
4th	VGG16	0.2	Adam	20	76%	96%
5th	VGG19	0.2	Adam	50	79%	99%
6th	CNN	0.3	Adam	100	85%	90%
7th	CNN	0.3	Adam	200	92%	96%
8th	VGG16	0.3	Adam	30	93%	97%
9th	VGG19	0.3	Adam	30	96%	98%

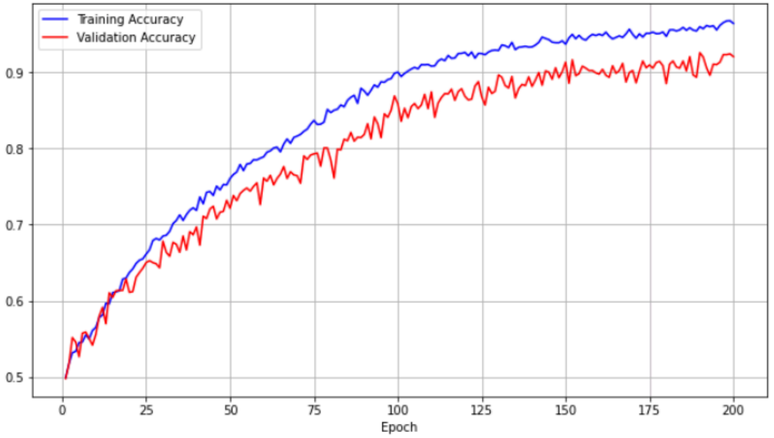
```python
class Net(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(3, 16, 3)
        self.conv2 = nn.Conv2d(16, 32, 3)
        self.conv3 = nn.Conv2d(32, 64, 3)
        self.conv4 = nn.Conv2d(64, 128, 3)
        self.conv5 = nn.Conv2d(128, 128, 3)
        self.pool = nn.MaxPool2d(2)
        self.flat = nn.Flatten()
        self.fc1 = nn.Linear(128 * 13 * 13, 1024)
        self.fc2 = nn.Linear(1024, 128)
        self.fc3 = nn.Linear(128, 4)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = self.pool(F.relu(self.conv3(x)))
        x = F.relu(self.conv4(x))
        x = F.relu(self.conv5(x))
        x = self.flat(x)
        x = F.leaky_relu(self.fc1(x))
        x = nn.Dropout(p=0.3)(x)
        x = F.leaky_relu(self.fc2(x))
        x = torch.softmax(self.fc3(x),dim=1)
        return x
```

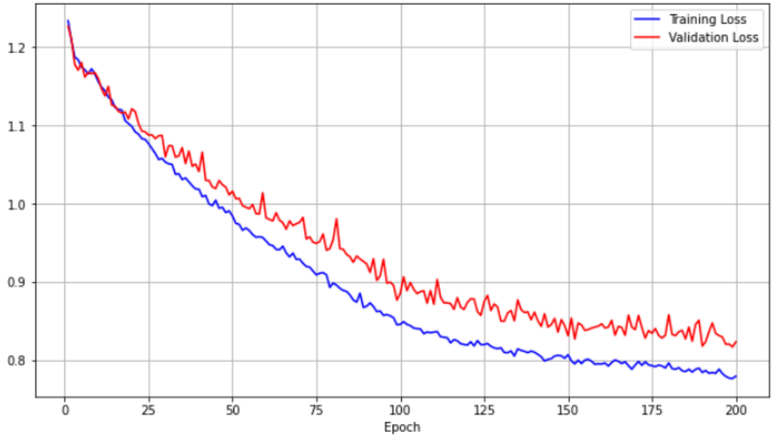




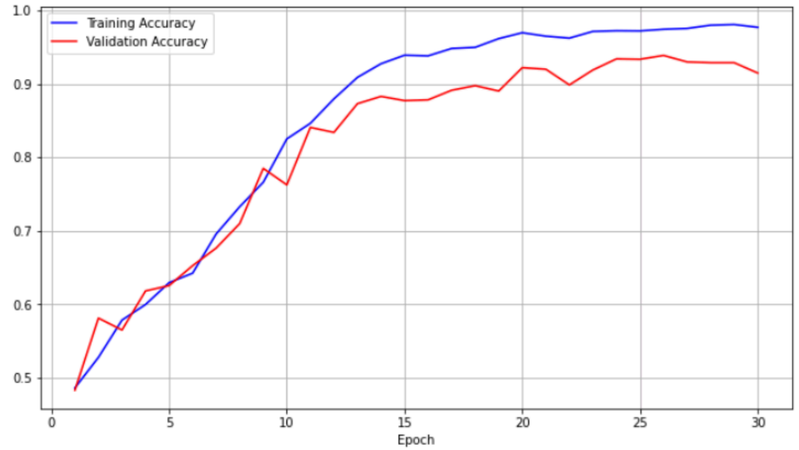


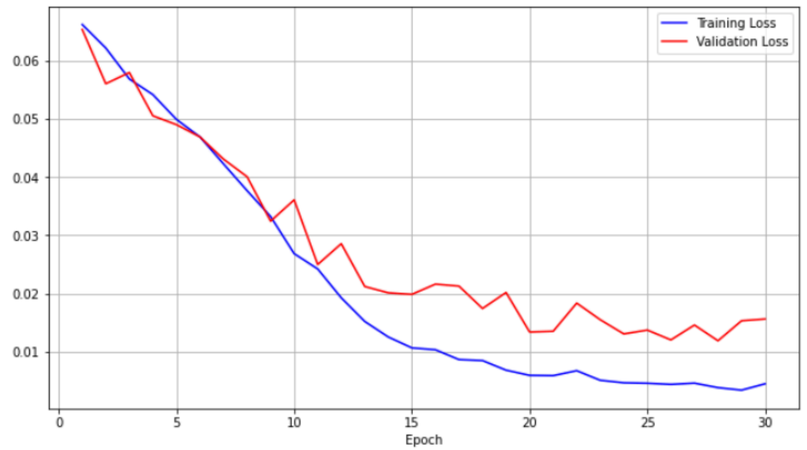
CNN Model - plots showing Accuracy and Loss for training and validation data (100 epochs)



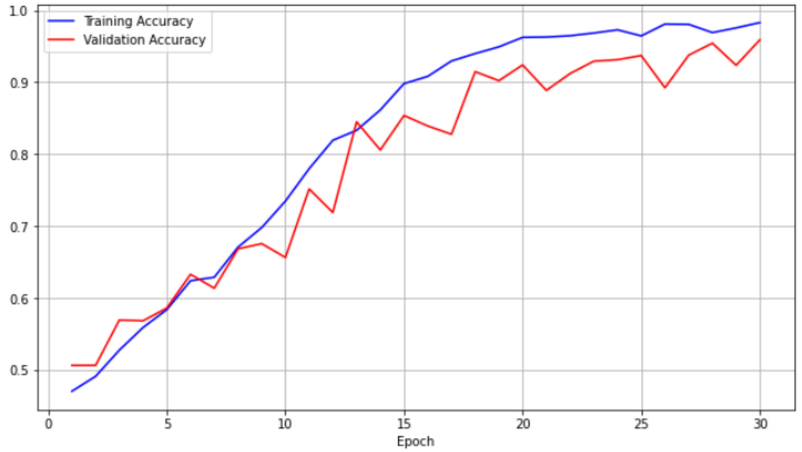


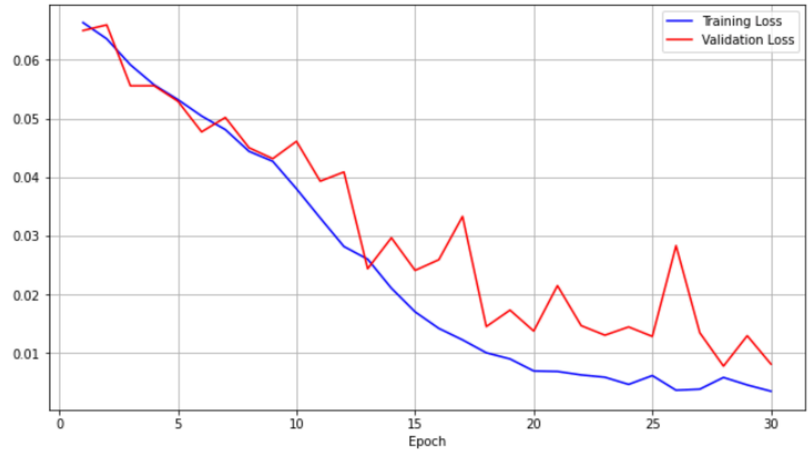
CNN Model - plots showing Accuracy and Loss for training and validation data (200 epochs)





VGG16 Model - plots showing Accuracy and Loss for training and validation data

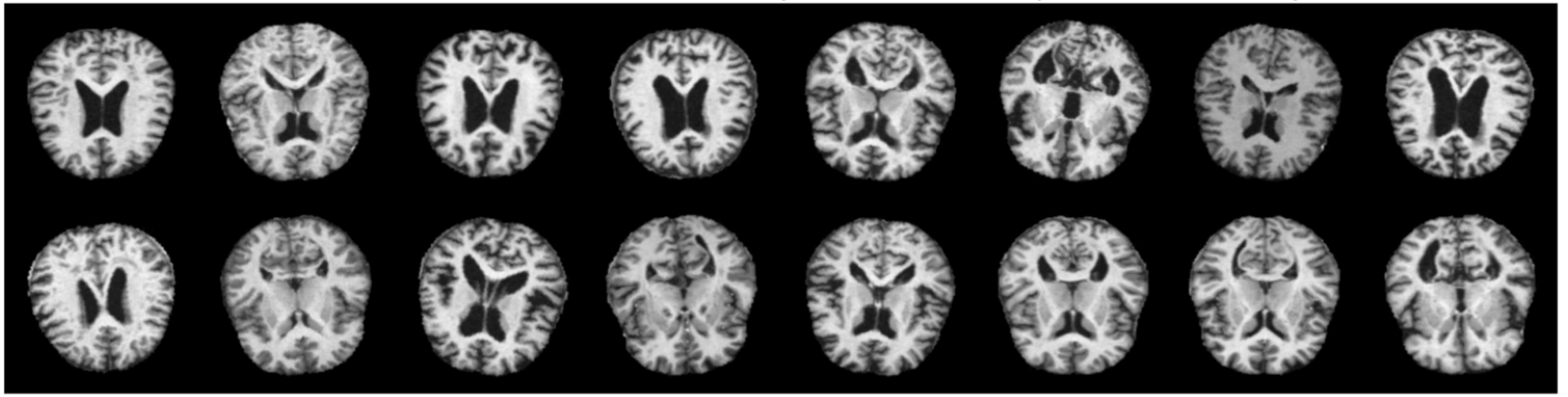




VGG19 Model - plots showing Accuracy and Loss for training and validation data

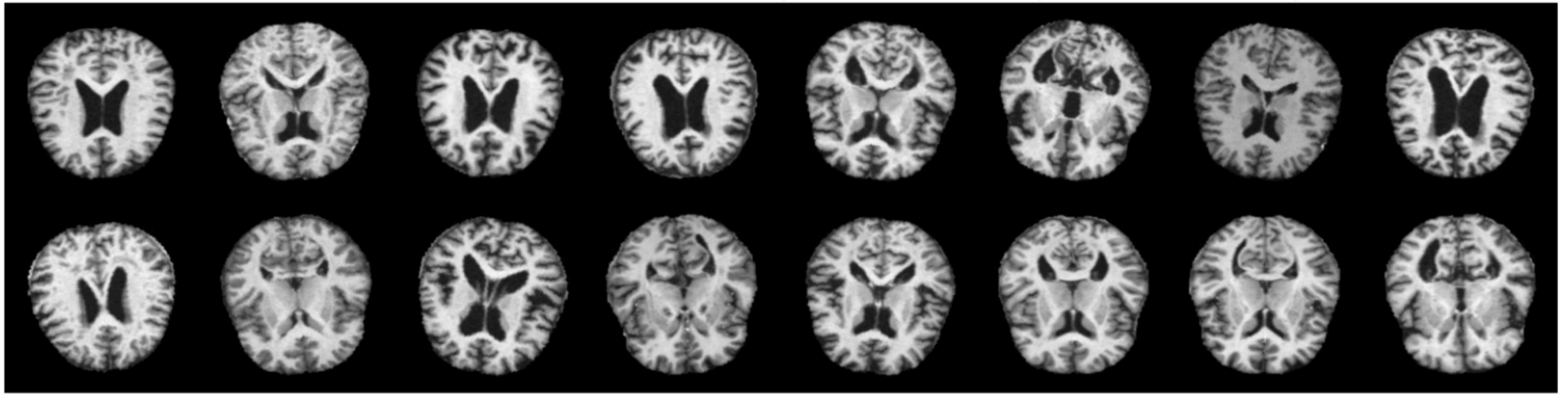
Ground truth:

['MildDemented', 'VeryMildDemented', 'NonDemented', 'MildDemented', 'MildDemented', 'NonDemented', 'NonDemented', 'MildDemented', 'VeryMildDemented', 'NonDemented', 'VeryMildDemented', 'NonDemented', 'VeryMildDemented', 'NonDemented', 'NonDemented', 'MildDemented']



Prediction:

['MildDemented', 'VeryMildDemented', 'MildDemented', 'MildDemented', 'MildDemented', 'NonDemented', 'NonDemented', 'MildDemented', 'VeryMildDemented', 'NonDemented', 'VeryMildDemented', 'NonDemented', 'VeryMildDemented', 'NonDemented', 'NonDemented', 'MildDemented']



VGG19 Predictions