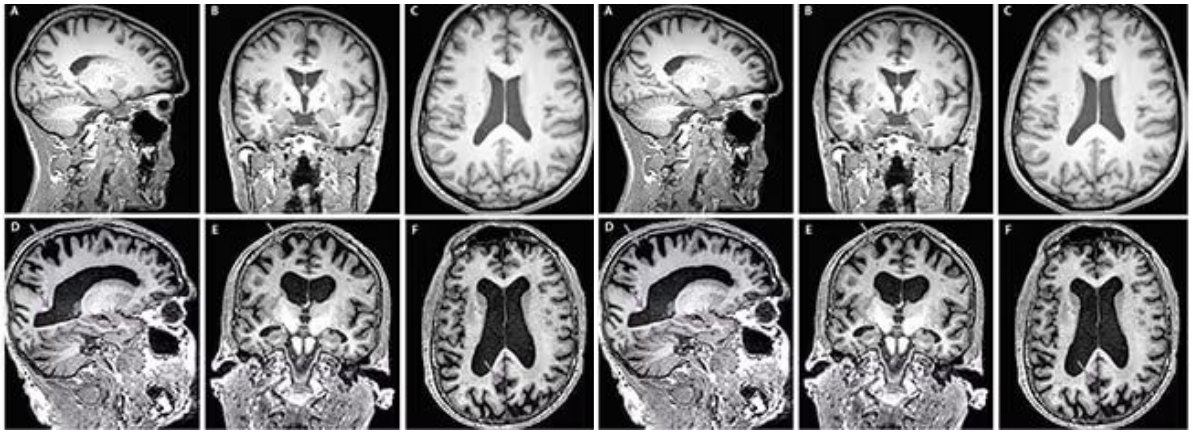
# fMRI Dementia Grader



By Ryan Clark and Syed Sabeeh Hassany

# Dementia Impact

**1 in 3**

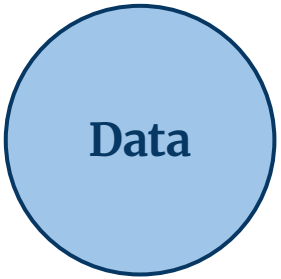
seniors dies with Alzheimer's or another dementia.

# Early diagnosis **helps!**

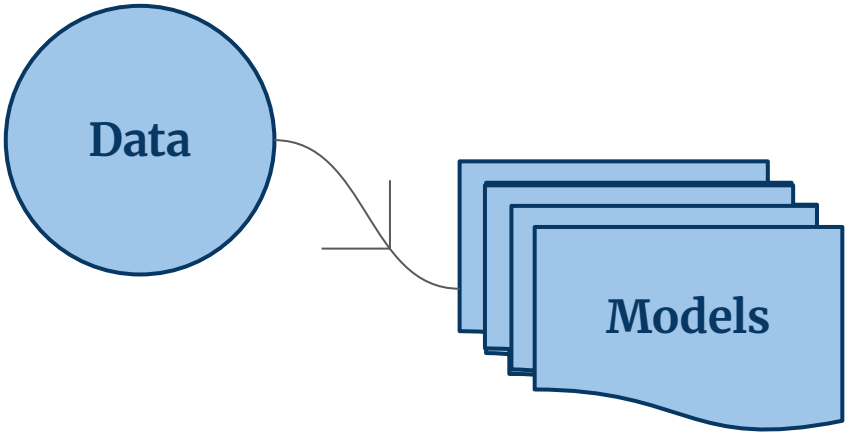
**1 in 3**

seniors dies with Alzheimer's or another dementia.

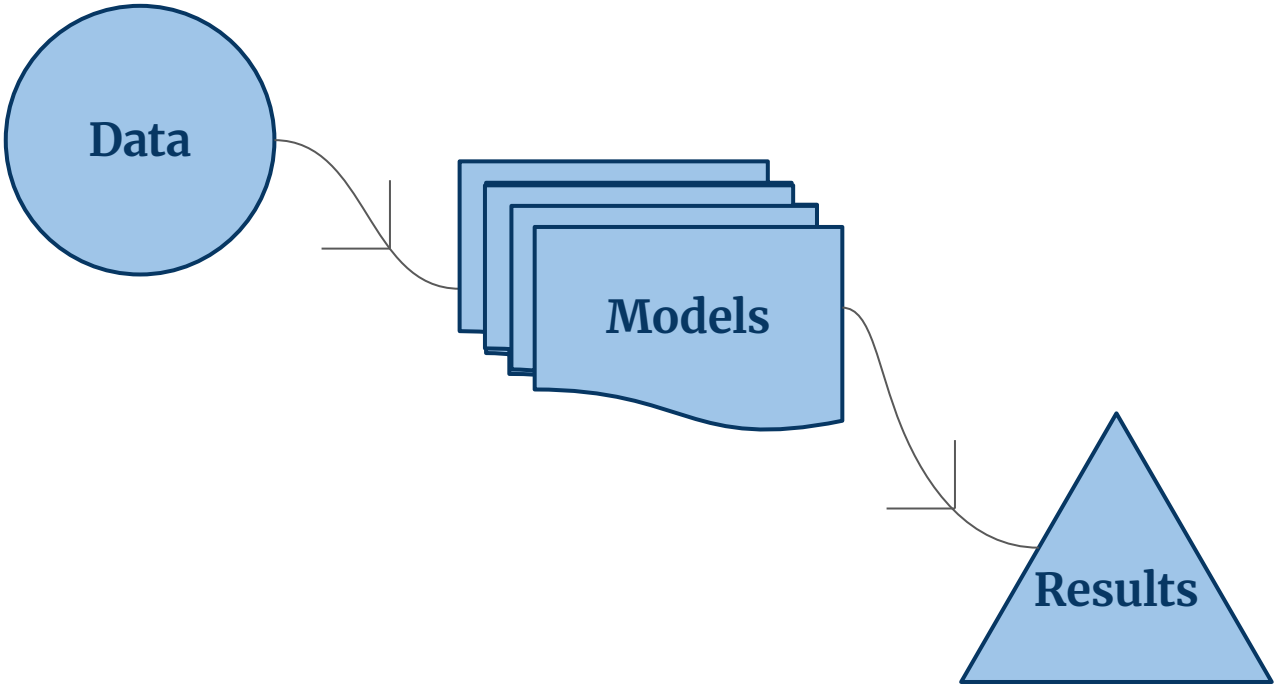
# Approach Overview



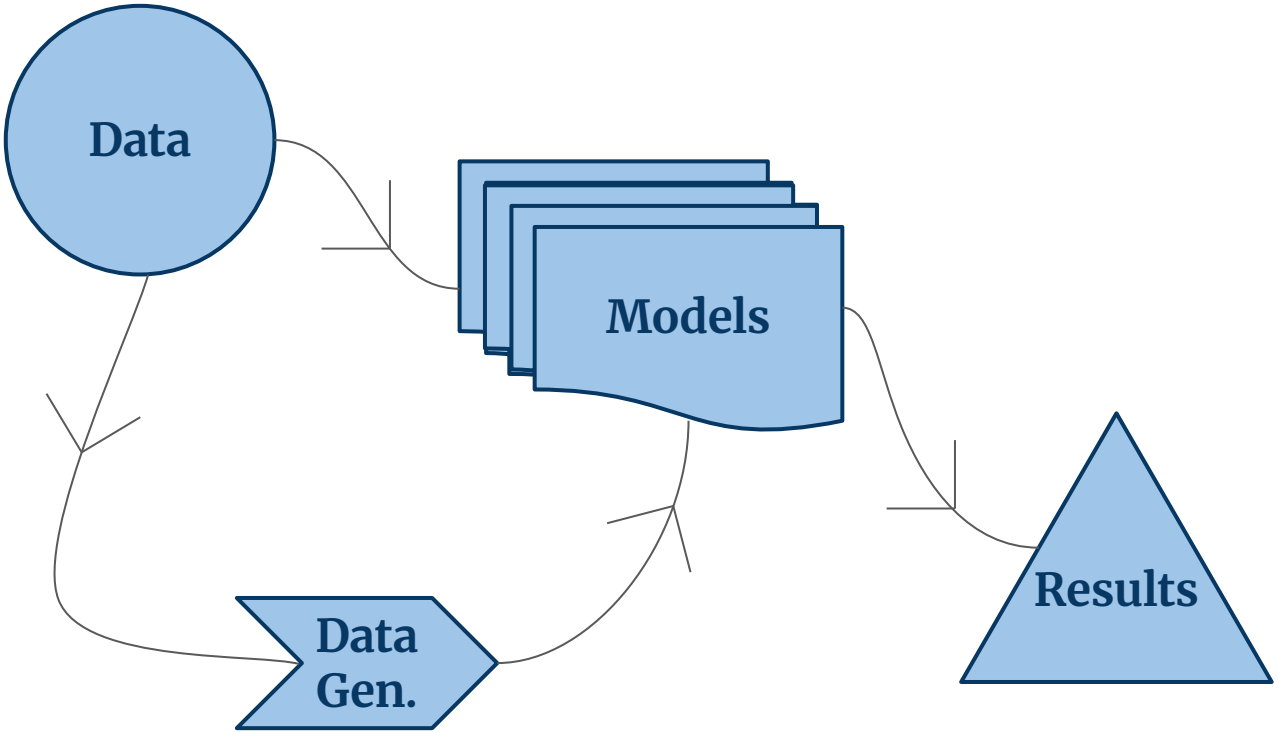
# Approach Overview



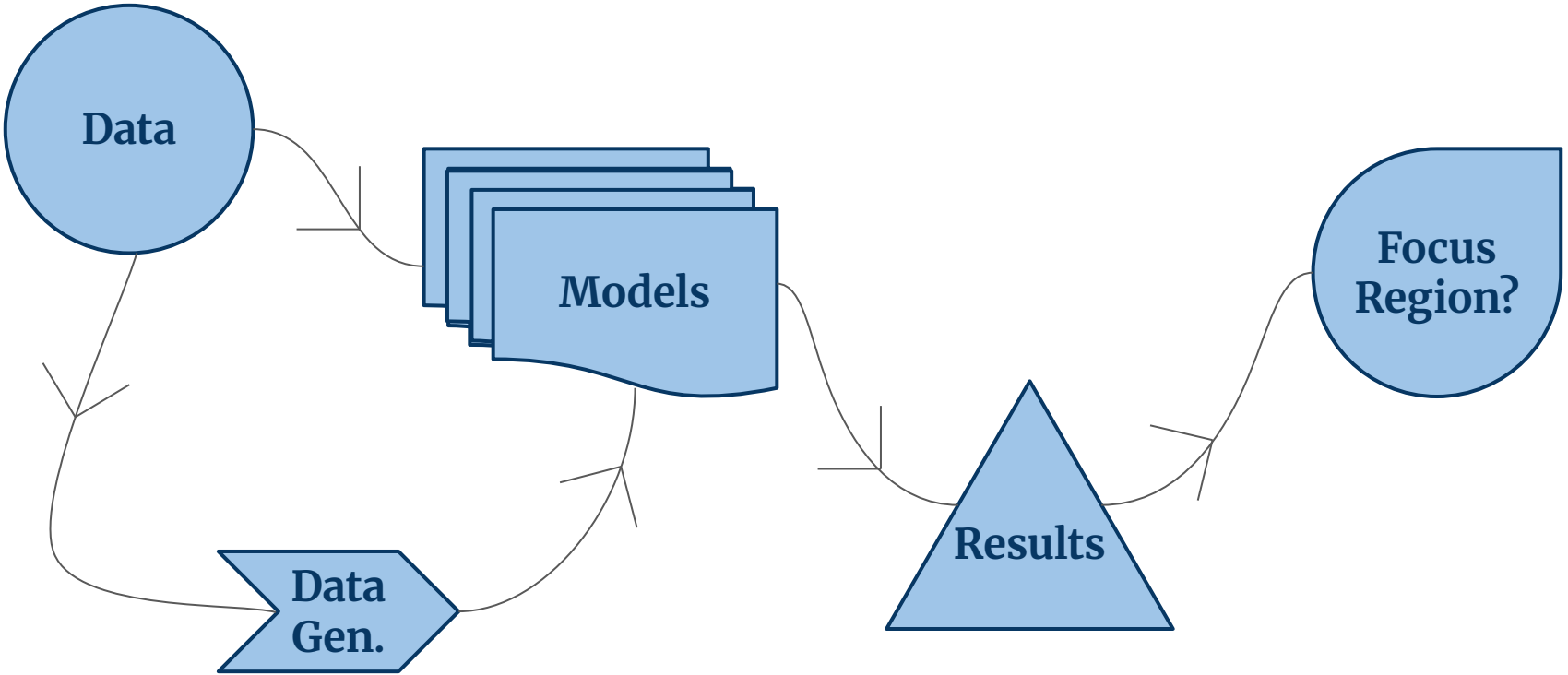
# Approach Overview



# Approach Overview



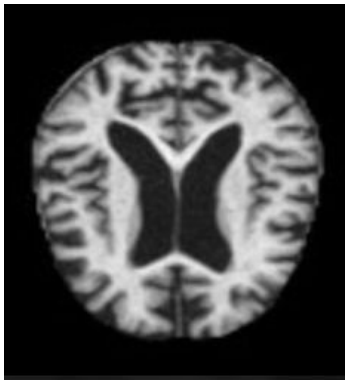
# Approach Overview



# Dataset

The data consists of **4 classes** of **preprocessed MRI images** collected from different repositories. There are a **total of 6400 images** and each image is **resized to 128 x 128 pixels**

- **Class - 1:** Mild Demented (896 images)
- **Class - 2:** Moderate Demented (64 images)
- **Class - 3:** Non Demented (3200 images)
- **Class - 4:** Very Mild Demented (2240 images)



# CNNs Used

**Resnet50** – Residual connections to enable the training of deeper neural networks for image classification tasks.

# CNNs Used

**Resnet50** – Residual connections to enable the training of deeper neural networks for image classification tasks.

**VGG16** – 16 layers featuring convolutional and max pooling layers, followed by fully connected layers.

# CNNs Used

**Resnet50** – Residual connections to enable the training of deeper neural networks for image classification tasks.

**VGG16** – 16 layers featuring convolutional and max pooling layers, followed by fully connected layers.

Convolutions with different kernel sizes and pooling operations to reduce dimensionality, resulting in improved accuracy – **Inception-V3**

# CNNs Used

**Resnet50** – Residual connections to enable the training of deeper neural networks for image classification tasks.

**VGG16** – 16 layers featuring convolutional and max pooling layers, followed by fully connected layers.

Convolutions with different kernel sizes and pooling operations to reduce dimensionality, resulting in improved accuracy – **Inception-V3**

Convolutional and max pooling layers followed by fully connected layers incorporating techniques to prevent overfitting. – **AlexNet 2D**

# CNNs Used

**Resnet50** – Residual connections to enable the training of deeper neural networks for image classification tasks.

**VGG16** – 16 layers featuring convolutional and max pooling layers, followed by fully connected layers.

Convolutions with different kernel sizes and pooling operations to reduce dimensionality, resulting in improved accuracy – **Inception-V3**

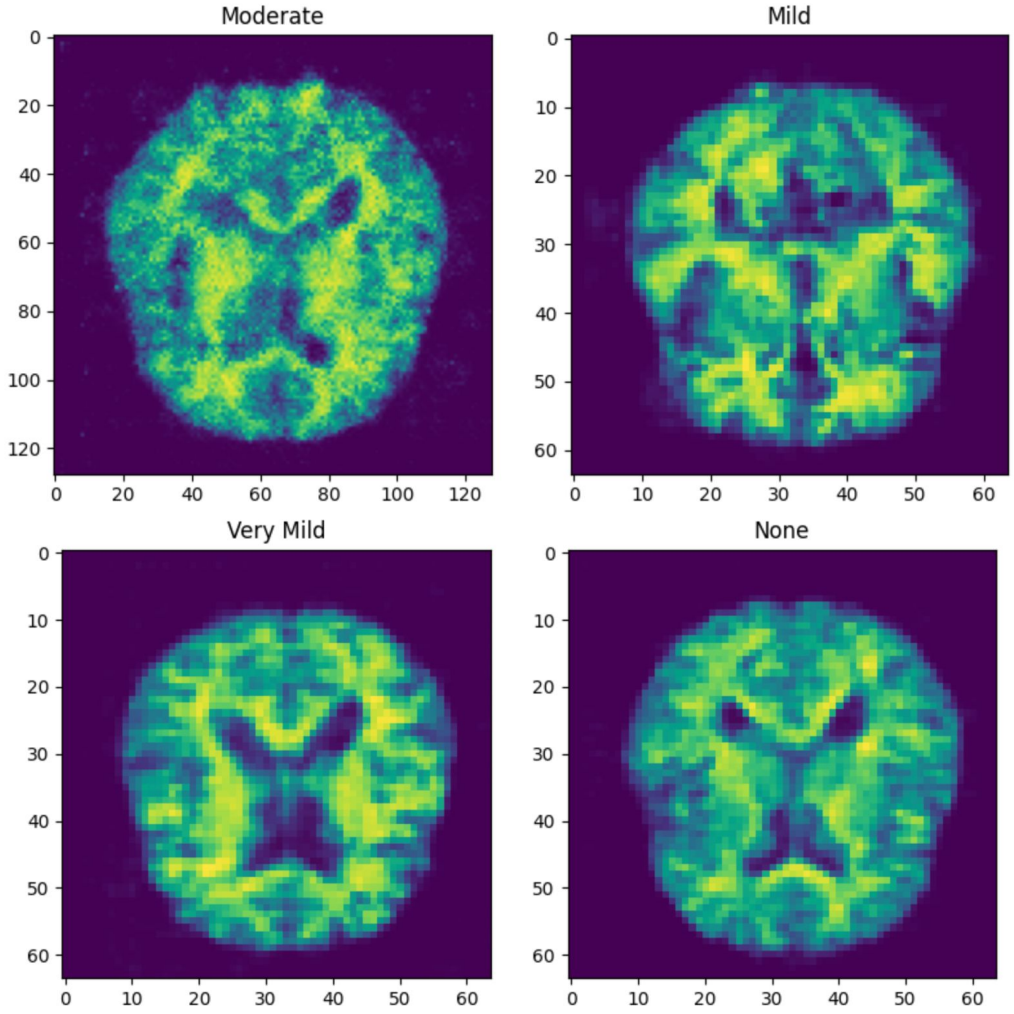
Convolutional and max pooling layers followed by fully connected layers incorporating techniques to prevent overfitting. – **AlexNet 2D**

# Example of Code

```python
class BasicBlock(nn.Module):
    expansion = 1
    def __init__(self, in_planes, planes, stride=1):
        super(BasicBlock, self).__init__()
        self.conv1 = nn.Conv2d(in_planes, planes, kernel_size=3, stride=stride, padding=1, bias=False)
        self.bn1 = nn.BatchNorm2d(planes)
        self.conv2 = nn.Conv2d(planes, planes, kernel_size=3, stride=1, padding=1, bias=False)
        self.bn2 = nn.BatchNorm2d(planes)

        self.shortcut = nn.Sequential()
        if stride != 1 or in_planes != self.expansion*planes:
            # make sure the shortcut has the same dimension
            self.shortcut = nn.Sequential(
                nn.Conv2d(in_planes, self.expansion*planes, kernel_size=1, stride=stride, bias=False),
                nn.BatchNorm2d(self.expansion*planes)
            )

    def forward(self, x):
        # finish the forward pass
        original = x

        out = self.conv1(x)
        out = self.bn1(out)
        out = F.relu(out)

        out = self.conv2(out)
        out = self.bn2(out)

        out += self.shortcut(original)
        out = F.relu(out)

```

# GAN Fake Images

- Using the same model we learned about in class, we trained each of our 4 datasets on it.
- Because of the limited size of our dataset, and the use of an outdated model we got a lot of repeat data within each batch.
- To the right is one example of each class of generated images.



# Model Results

| Model \ Perf. Accuracy | Normal Data | Fake Data |
|---|---|---|
| Resnet | 93.31 % | 56.25% |
| VGG16 | 46.18 % | 23.32% |
| Inception - V3 | 89.80 % | 46.86% |
| AlexNet | 92.30 % | 62.50% |

# Model Results

| Model \ Perf. Accuracy | Normal Data | Fake Data |
|---|---|---|
| Resnet | 93.31 % | 56.25% |
| VGG16 | 46.18 % | 23.32% |
| Inception - V3 | 89.80 % | 46.86% |
| AlexNet | 92.30 % | 62.50% |

# Model Results

| Model \ Perf. Accuracy | Normal Data | Fake Data |
|---|---|---|
| Resnet | 93.31 % | 56.25% |
| VGG16 | 46.18 % | 23.32% |
| Inception - V3 | 89.80 % | 46.86% |
| AlexNet | 92.30 % | 62.50% |

# Model Results

| Model \ Perf. Accuracy | Normal Data | Fake Data |
|---|---|---|
| Resnet | 93.31 % | 56.25% |
| VGG16 | 46.18 % | 23.32% |
| Inception - V3 | 89.80 % | 46.86% |
| AlexNet | 92.30 % | 62.50% |

# 1. Save model with checkpoints.

- Initially we did not save our checkpoints
- After saving checkpoint we did not have to run our model each time

# Challenges and Takeaways

1. Save model with checkpoints.
   - Initially we did not save our checkpoints
   - After saving checkpoint we did not have to run our model each time
2. Take advantage of online resources.
   - Initially we tried to code models from scratch
   - After using online resources we were able to gain momentum

# Challenges and Takeaways

1. Save model with checkpoints.
   - Initially we did not save our checkpoints
   - After saving checkpoint we did not have to run our model each time
2. Take advantage of online resources.
   - Initially we tried to code models from scratch
   - After using online resources we were able to gain momentum
3. Do what you know best.
   - Initially we tried to use find models we did not use in class
   - After implementing class models we had a strong foundation to start from

# Challenges and Takeaways

## 1. Save model with checkpoints.

- Initially we did not save our checkpoints
- After saving checkpoint we did not have to run our model each time

## 2. Take advantage of online resources.

- Initially we tried to code models from scratch
- After using online resources we were able to gain momentum

## 3. Do what you know best.

- Initially we tried to use find models we did not use in class
- After implementing class models we had a strong foundation to start from

# 1. Identify high impact regions.

- Use pixel analysis or GrabCAM to find areas with biggest impact
- Identify areas of focus for assistant physicians + future treatments

# Next Steps

1. Identify high impact regions.
   - Use pixel analysis or GradCAM to find areas with biggest impact
   - Identify areas of focus for assistant physicians + future treatments
2. Validate models on larger dataset
   - Deploy our models on more fMRI diagnosis to validate performance
   - Identify potential overfitting of data

# Next Steps

1. Identify high impact regions.
   - Use pixel analysis or GradCAM to find areas with biggest impact
   - Identify areas of focus for assistant physicians + future treatments
2. Validate models on larger dataset
   - Deploy our models on more fMRI diagnosis to validate performance
   - Identify potential overfitting of data
3. Improve Fake Image Generation
   - Use StyleGAN3, a newer and better generation model to create better data.
   - We can add our these to our existing dataset to accomplish goal #2

# The End

Thank you!