DECODING ALZHEIMER'S: MULTI-CLASS CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS

Aviral Asthana
B.Tech (CSE with specialization in AIML)

OVERVIEW

01 What is Alzheimer's?

02 Symptoms

03 Existing Methods

04 What is CNN?

05 How can CNN help?

06 Code

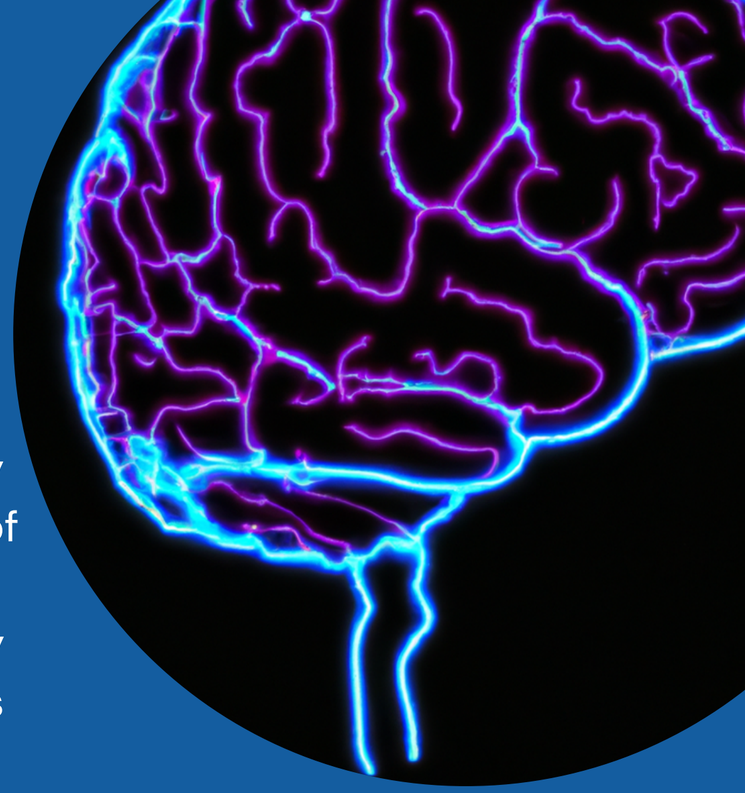
07 Output

08 Charts

09 Research Paper Ref. and Resources used

10 Conclusion and Contribution

WHAT IS ALZHEIMER'S?



- Alzheimer's disease is a progressive and irreversible brain disorder that primarily affects memory, cognitive abilities, and behavior. It is the most common cause of dementia in elderly individuals.
- The exact cause of Alzheimer's is not fully understood, but it is characterized by the abnormal accumulation of two types of protein in the brain: amyloid plaques and tau tangles.
- These protein deposits disrupt the communication between brain cells and lead to their gradual degeneration and death.
- As the disease progresses, individuals with Alzheimer's experience memory loss, confusion, difficulty in problem-solving, language difficulties, and changes in personality and behavior. In the later stages, they may become unable to perform everyday tasks and eventually require round-the-clock care.

SYMPTOMS

The symptoms of Alzheimer's disease can vary from person to person, and they typically develop slowly and worsen over time. Some of the common symptoms include:

01 Memory Loss: Difficulty in remembering recent events, names, or appointments. People may ask for the same information repeatedly.

02 Cognitive Decline: Trouble with thinking, problem-solving, and decision-making. Reduced ability to plan and organize.

03 Language Difficulties: Struggling to find the right words, following conversations, or understanding written and spoken language.

04 Disorientation: Getting lost in familiar places, not recognizing surroundings, or losing track of time.

05 Mood and Behavior Changes: Frequent mood swings, increased anxiety, depression, irritability, or withdrawal from social activities.

06 Trouble Completing Familiar Tasks: Difficulty performing tasks that were once familiar, such as cooking, managing finances, or using household appliances.

07 Poor Judgment: Exhibiting poor judgment and making questionable decisions, including falling for scams or neglecting personal safety.

08 Personality Changes: Undergoing significant personality changes, becoming confused, suspicious, or fearful.

EXISTING METHODS

- Treatment Techniques for Alzheimer's Disease:
  - Medications
  - Behavioral Interventions
  - Occupational Therapy
  - Physical Exercise
  - Nutrition
  - Support and Education for Caregivers
  - Safety Measures
  - Clinical Trials

WHAT IS CONVOLUTIONAL NEURAL NETWORK?

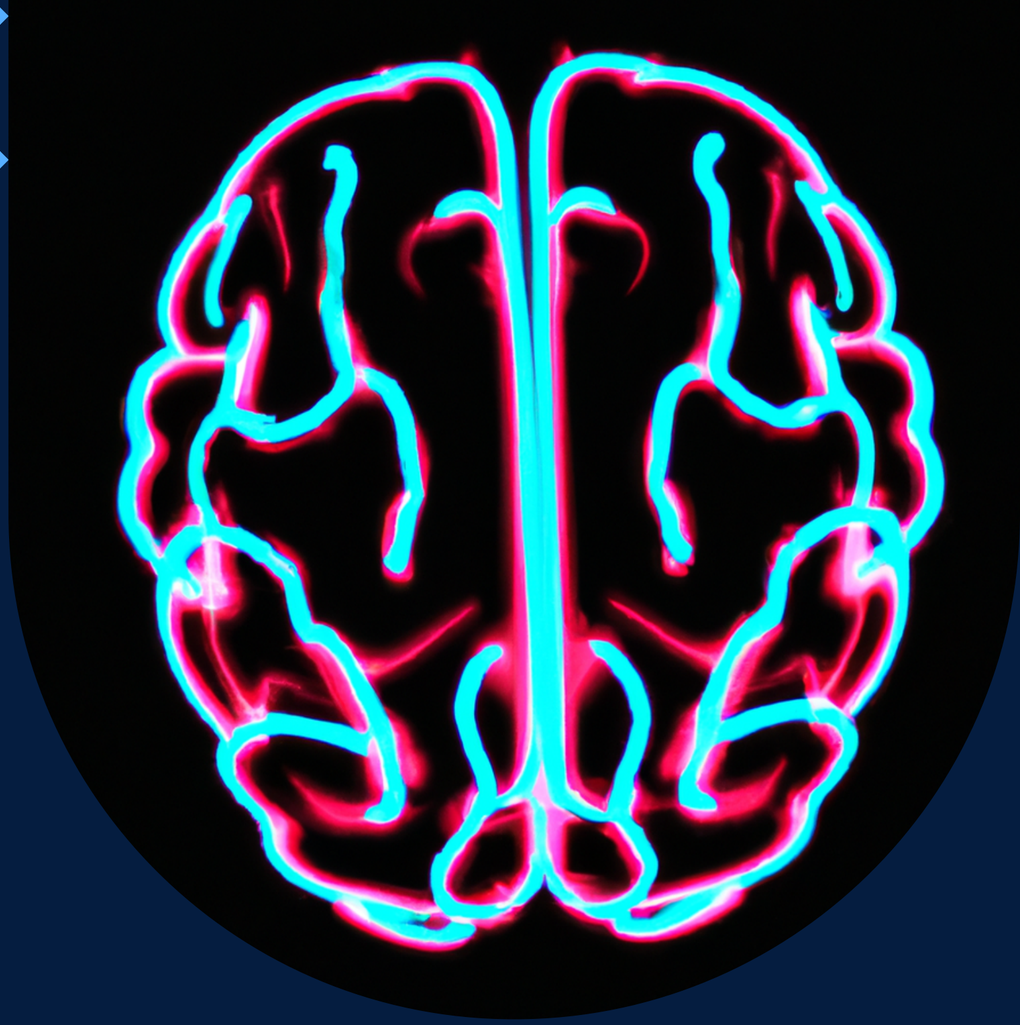
- A Convolutional Neural Network (CNN) is a specialized type of deep neural network designed for processing and analyzing visual data, such as images and videos. It has revolutionized the field of computer vision by significantly improving the accuracy and efficiency of image recognition tasks.
- The architecture of Convolutional Neural Networks consists of several layers, Including Convolution, Activation (ReLU) Layer, Pooling, Flattening layer, and Fully connected layer

How can CNN help?

- Convolutional Neural Networks (CNNs) can be instrumental in Alzheimer's detection by analyzing brain imaging data, such as MRI or PET scans, and automatically extracting relevant features from images.
- They can differentiate between normal brain regions and areas affected by Alzheimer's-related changes, aiding in early detection and tracking disease progression over time. CNNs can provide valuable insights to healthcare professionals, assisting in accurate diagnosis, risk prediction, and treatment planning.
- However, they should be used as an aid to medical professionals rather than a replacement, and rigorous validation and ethical considerations are crucial to ensure their responsible application in healthcare.

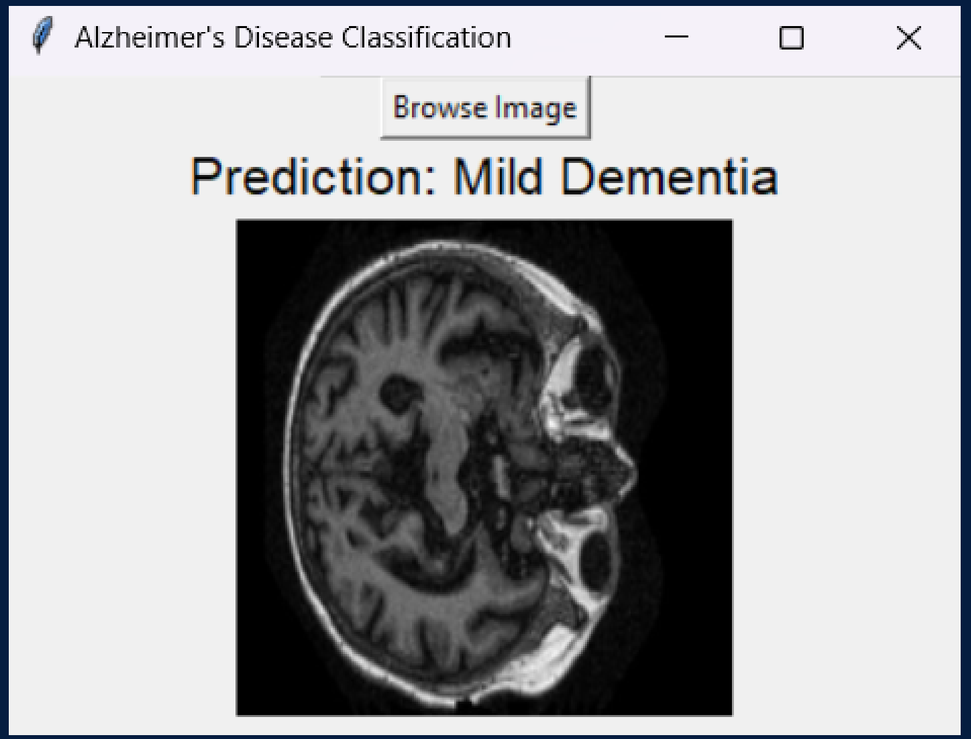
CODE

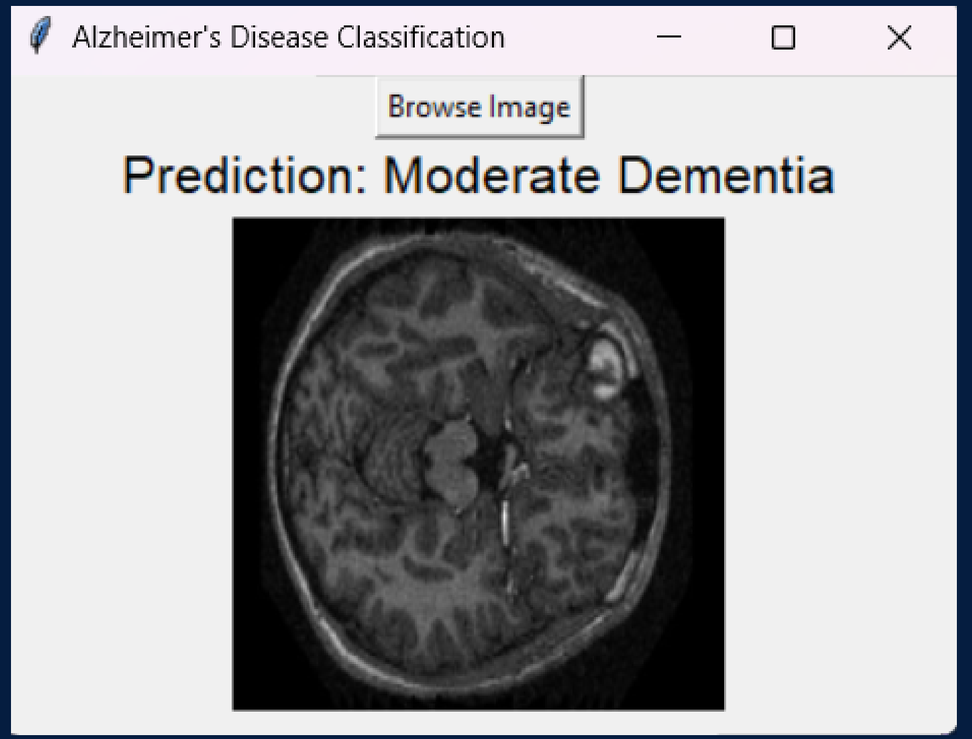
```
from keras.backend import dropout
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from keras.regularizers import l2

train_data_dir = 'C:/Users/avi11/Downloads/archive/Train Data'
test_data_dir = 'C:/Users/avi11/Downloads/archive/Test Data'

image_height, image_width = 150, 150
batch_size = 32

train_datagen = ImageDataGenerator(
    rescale=1.0/255.0,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)

test_datagen = ImageDataGenerator(rescale=1.0/255.0)
```



```
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(image_height, image_width),
    batch_size=batch_size,
    class_mode='sparse'
)

test_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=(image_height, image_width),
    batch_size=batch_size,
    class_mode='sparse'
)
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(image_height, image_width, 3)),
    MaxPooling2D((2, 2)),

    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),

    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),

    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),

    Conv2D(256, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),

    Flatten(),
    Dense(256, activation='relu'),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(4, activation='softmax')
])
```

```
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

epochs = 30
model.fit(train_generator, epochs=epochs)
```





OUTPUT

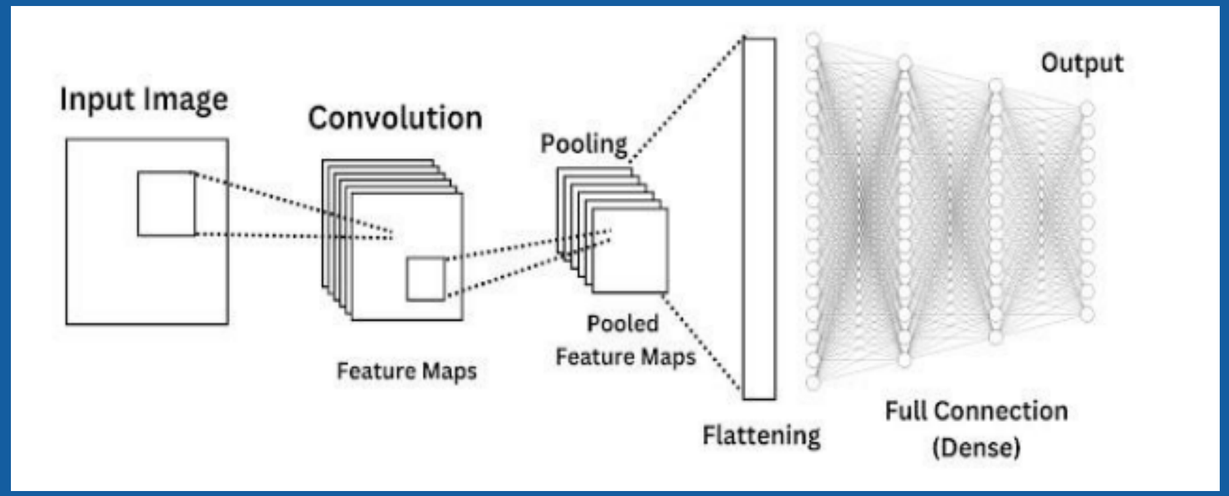
```
125/125 [==============================] - 87s 693ms/step - loss: 0.5023 - accuracy: 0.8123
Epoch 14/30
125/125 [==============================] - 87s 691ms/step - loss: 0.4287 - accuracy: 0.8371
Epoch 15/30
125/125 [==============================] - 86s 688ms/step - loss: 0.3774 - accuracy: 0.8591
Epoch 16/30
125/125 [==============================] - 86s 687ms/step - loss: 0.3427 - accuracy: 0.8734
Epoch 17/30
125/125 [==============================] - 84s 673ms/step - loss: 0.2915 - accuracy: 0.8992
Epoch 18/30
125/125 [==============================] - 85s 678ms/step - loss: 0.2829 - accuracy: 0.8977
Epoch 19/30
125/125 [==============================] - 87s 694ms/step - loss: 0.2495 - accuracy: 0.9078
Epoch 20/30
125/125 [==============================] - 90s 718ms/step - loss: 0.2395 - accuracy: 0.9203
Epoch 21/30
125/125 [==============================] - 88s 701ms/step - loss: 0.2192 - accuracy: 0.9188
Epoch 22/30
125/125 [==============================] - 93s 739ms/step - loss: 0.1806 - accuracy: 0.9383
Epoch 23/30
125/125 [==============================] - 91s 721ms/step - loss: 0.1952 - accuracy: 0.9306
Epoch 24/30
125/125 [==============================] - 89s 708ms/step - loss: 0.1701 - accuracy: 0.9439
Epoch 25/30
125/125 [==============================] - 93s 745ms/step - loss: 0.1641 - accuracy: 0.9486
Epoch 26/30
125/125 [==============================] - 95s 763ms/step - loss: 0.1828 - accuracy: 0.9358
Epoch 27/30
125/125 [==============================] - 104s 830ms/step - loss: 0.1600 - accuracy: 0.9491
Epoch 28/30
125/125 [==============================] - 126s 1s/step - loss: 0.1600 - accuracy: 0.9439
Epoch 29/30
125/125 [==============================] - 96s 765ms/step - loss: 0.1324 - accuracy: 0.9574
Epoch 30/30
125/125 [==============================] - 86s 685ms/step - loss: 0.1607 - accuracy: 0.9466
```

The conclusion of the provided code is that the model has been trained on the training dataset for 30 epochs using the specified CNN architecture and data augmentation techniques. After training, the model's performance was evaluated on the test dataset, and the test loss and test accuracy were calculated and printed to the console

VISUALIZING THE NEURAL NETWORK,

```
plot_model(model, to_file='model_graph.png', show_shapes=True)
```

✓ 0.7s



Structure of the CNN

conv2d_input	input:	[(None, 150, 150, 3)]
InputLayer	output:	[(None, 150, 150, 3)]

conv2d	input:	(None, 150, 150, 3)
Conv2D	output:	(None, 148, 148, 32)

max_pooling2d	input:	(None, 148, 148, 32)
MaxPooling2D	output:	(None, 74, 74, 32)

conv2d_1	input:	(None, 74, 74, 32)
Conv2D	output:	(None, 72, 72, 64)

max_pooling2d_1	input:	(None, 72, 72, 64)
MaxPooling2D	output:	(None, 36, 36, 64)

conv2d_2	input:	(None, 36, 36, 64)
Conv2D	output:	(None, 34, 34, 128)

max_pooling2d_2	input:	(None, 34, 34, 128)
MaxPooling2D	output:	(None, 17, 17, 128)

conv2d_3	input:	(None, 17, 17, 128)
Conv2D	output:	(None, 15, 15, 128)

max_pooling2d_3	input:	(None, 15, 15, 128)
MaxPooling2D	output:	(None, 7, 7, 128)

conv2d_4	input:	(None, 7, 7, 128)
Conv2D	output:	(None, 5, 5, 256)

max_pooling2d_4	input:	(None, 5, 5, 256)
MaxPooling2D	output:	(None, 2, 2, 256)

flatten	input:	(None, 2, 2, 256)
Flatten	output:	(None, 1024)

dense	input:	(None, 1024)
Dense	output:	(None, 256)

dense_1	input:	(None, 256)
Dense	output:	(None, 128)

dropout	input:	(None, 128)
Dropout	output:	(None, 128)

dense_2	input:	(None, 128)
Dense	output:	(None, 4)

Existing Research Paper References:

Convolutional neural network (CNN), a class of artificial neural networks that has become dominant in various computer vision tasks, is attracting interest across a variety of domains, including radiology. CNN is designed to automatically and adaptively learn spatial hierarchies of features through backpropagation by using multiple building blocks, such as convolution layers, pooling layers, and fully connected layers

https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9

Alzheimer's Disease is predicted using ML algorithms by using a feature selection and extraction technique, and the classification is conducted based on the oasis longitudinal dataset. The different techniques involved in analyzing brain images for diagnosing diseases of the brain to provide a brief overview.

https://www.frontiersin.org/articles/10.3389/fpubh.2022.853294/full

Software and Hardware used

Hardware-

- Intel core i5-10300H CPU @ 2.50GHz
- RAM - 8.00 GB
- NVIDIA GeForce 1650 Ti

Software and other Resources-

- Dataset - Kaggle OASIS Alzheimer's Detection
- Visual Studio Code
- Python Libraries used-
  - Keras
    - backend
    - preprocessing.image
    - models
    - layers
    - utils

THANK YOU!