Data Collection and Preprocessing Phase

Date	7 July 2024
Team ID	SWTID1720426301
Project Title	Cognitive Care: Early Intervention for Alzheimer's Disease
Maximum Marks	6 Marks

Data Exploration and Preprocessing Report

The data exploration and preprocessing approach for Alzheimer's Disease involves a series of steps including resizing, normalizing, augmenting, denoising, contrast adjustment, edge detection, color space conversion, cropping, batch normalization, and data whitening. These methods collectively enhance data quality, facilitate model generalization, and optimize neural network training, essential for achieving accurate and robust Alzheimer's Disease diagnosis through advanced computer vision techniques.

Section	**Description**
Data Overview	The project utilizes Alzheimer's MRI scans sourced from Kaggle, categorized into four classes.
Resizing	Resizing to 180x180 pixels ensures consistent input dimensions for the Xception model as per the code's preprocessing requirements.
Normalization	Normalization in the code involves scaling pixel values to a range between 0 and 1, ensuring consistent data range across images to facilitate effective model training and convergence.
Data Augmentation	Data augmentation in the code (brightness adjustment, zooming, horizontal flipping) diversifies training data, boosting model robustness in Alzheimer's Disease pattern recognition.
Data Balancing	Data balancing in the code uses SMOTE to address class imbalance by generating synthetic samples, improving model accuracy across all Alzheimer's Disease categories.

<table>
<tr><td>Transfer Learning</td><td>Transfer learning in the code uses a pre-trained Xception model with frozen layers to improve Alzheimer's Disease classification performance.</td></tr>
<tr><td>Batch Normalization</td><td>Batch normalization in the code stabilizes training by normalizing input batches, enhancing model convergence and generalization.</td></tr>
<tr><td colspan="2">Data Preprocessing Code Screenshots</td></tr>
<tr><td>Loading Data</td><td><pre>trainPath = r"/content/Alzheimer_s Dataset/train"
testPath = r"/content/Alzheimer_s Dataset/test"</pre></td></tr>
<tr><td>Resizing</td><td><pre>from tensorflow.keras.preprocessing.image import ImageDataGenerator as IDG
IMG_SIZE = 180
IMAGE_SIZE = [180, 180]
DIM = (IMG_SIZE, IMG_SIZE)</pre></td></tr>
<tr><td>Normalization</td><td><pre>ZOOM = [.99, 1.01]
BRIGHT_RANGE = [0.8, 1.2]
HORZ_FLIP = True
FILL_MODE="constant"
DATA_FORMAT = "channels_last"
WORK_DIR="/content/Alzheimer_s Dataset/train"
work_dr = IDG(rescale = 1./255,
              brightness_range=BRIGHT_RANGE,
              zoom_range=ZOOM,
              data_format=DATA_FORMAT,
              fill_mode=FILL_MODE,
              horizontal_flip=HORZ_FLIP)</pre></td></tr>
<tr><td>Data Augmentation</td><td><pre>train_data_gen = work_dr.flow_from_directory(directory=WORK_DIR,
                                             target_size=DIM,
                                             batch_size=6500,
                                             shuffle=False)</pre></td></tr>
</table>

Data Balancing	```train_data, train_labels = train_data_gen.next()```
	```# before oversampling```
	```print(train_data.shape, train_labels.shape)```
	```(5121, 180, 180, 3) (5121, 4)```
	```from imblearn.over_sampling import SMOTE```
	```sm = SMOTE(random_state=42)```
	```train_data, train_labels = sm.fit_resample(train_data.reshape(-1, IMG_SIZE * IMG_SIZE * 3), train_labels)```
	```train_data = train_data.reshape(-1, IMG_SIZE, IMG_SIZE, 3)```
	```print(train_data.shape, train_labels.shape)```

Transfer Learning	```xcep_model = Xception(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)```
	```for layer in xcep_model.layers:```
	``` layer.trainable = False```

Batch Normalization	```from tensorflow.keras.models import Sequential```
	```from tensorflow.keras.layers import SeparableConv2D, BatchNormalization, GlobalAveragePooling2D , Dropout```
	```custom_inception_model = Sequential([```
	```xcep_model,```
	```Dropout(0.5),```
	```GlobalAveragePooling2D(),```
	```Flatten(),```
	```BatchNormalization(),```
	```Dense (512, activation ='relu'),```
	```BatchNormalization(),```
	```Dropout(0.5),```
	```Dense (256, activation ='relu'),```
	```BatchNormalization(),```
	```Dropout(0.5),```
	```Dense (128, activation ='relu'),```
	```BatchNormalization(),```
	```Dropout(0.5),```
	```Dense (64, activation='relu'),```
	```Dropout(0.5),```
	```BatchNormalization(),```
	```Dense(4, activation='softmax')```
	```], name = "inception_cnn_model")```