Data Collection and Preprocessing Phase

Date	19 July 2024
Team ID	SWTID1720627211
Project Title	Cognitive Care: Early Intervention for Alzheimer’s Disease
Maximum Marks	6 Marks

Data Exploration and Preprocessing Template

The process of data exploration and preprocessing for Alzheimer's Disease involves several key steps. These include resizing images, normalizing them, augmenting the data, denoising, adjusting contrast, performing edge detection, converting color spaces, cropping, applying batch normalization, and whitening the data. These techniques collectively improve data quality, aid in model generalization, and optimize neural network training. This comprehensive approach is crucial for achieving accurate and robust Alzheimer's Disease diagnosis using advanced computer vision methods.

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.

Transfer Learning	Transfer learning in the code uses a pre-trained Xception model with frozen layers to improve Alzheimer's Disease classification performance.
Batch Normalization	Batch normalization in the code stabilizes training by normalizing input batches, enhancing model convergence and generalization.

Data Preprocessing Code Screenshots

Loading Data

```
[ ]  trainPath = r"/content/Alzheimer_s Dataset/train"
     testPath = r"/content/Alzheimer_s Dataset/test"
```

Resizing

```
[ ]  from tensorflow.keras.preprocessing.image import ImageDataGenerator as IDG

     IMG_SIZE = 180
     IMAGE_SIZE = [180, 180]
     DIM = (IMG_SIZE, IMG_SIZE)
```

Normalization

```
ZOOM = [0.99, 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)
```

Data Augmentation

```
train_data_gen = work_dr.flow_from_directory(WORK_DIR, target_size=DIM,
                                             batch_size=6500, shuffle=False)
```

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)`
	`(10240, 180, 180, 3) (10240, 4)`

Transfer Learning	`xcep_model = Xception(input_shape= IMAGE_SIZE + [3],`
	`weights='imagenet', include_top=False)`
	`Downloading data from https://storage.googleapis.com/tensorflow/kera`
	`83683744/83683744 [==============================] - 1s 0us/step`

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")`