**Assignment 2**

**1. Introduction**

Face recognition is a widely studied problem in the field of computer vision and machine learning. This assignment demonstrates the use of a Convolutional Neural Network (CNN) to distinguish between two selected people from the Labeled Faces in the Wild (LFW) dataset.

By training a CNN on grayscale images of faces, the system learns to classify input images into one of two categories. This assignment highlights key deep learning concepts, including data preprocessing, data augmentation, CNN architecture, model training, and evaluation.

**2. Dataset**

* Dataset Used: Labeled Faces in the Wild (LFW)
* Selection: Only persons with at least 20 images were considered. From these, two people were chosen.
* Preprocessing:
  + Images resized to 100×100 pixels.
  + Converted to grayscale (1 channel).
  + Normalized pixel values to the range [0,1].
* Dataset Split:
  + Training set: 80% of images
  + Test set: 20% of images

**3. Data Augmentation**

To increase model robustness and prevent overfitting, ImageDataGenerator was used for augmentation with the following transformations:

* Random rotation (±20°)
* Zoom (up to 15%)
* Horizontal shifts and vertical shifts (20%)
* Horizontal flipping

This helps the model generalize to variations in pose, scale, and orientation.

**4. Model Architecture**

A Sequential CNN model was constructed with the following layers:

1. Conv2D (32 filters, 3×3, ReLU) → feature extraction
2. MaxPooling2D (2×2) → reduces spatial dimensions
3. Conv2D (64 filters, 3×3, ReLU)
4. MaxPooling2D (2×2)
5. Conv2D (128 filters, 3×3, ReLU)
6. MaxPooling2D (2×2)
7. Flatten → converts 2D features into 1D
8. Dense (128 units, ReLU) → fully connected layer
9. Dropout (0.5) → prevents overfitting
10. Dense (1 unit, Sigmoid) → outputs probability for binary classification

**5. Compilation**

The model was compiled with:

* Optimizer: Adam (adaptive learning rate optimization)
* Loss Function: Binary Crossentropy (for two-class classification)
* Metric: Accuracy

**6. Training**

The model was trained for 20 epochs using a batch size of 32.

* Training Data: Augmented using ImageDataGenerator
* Validation Data: Raw test set

During training, accuracy and loss were recorded for both training and validation sets.

A sample code snippet for training:

history = model.fit(datagen.flow(X\_train, y\_train, batch\_size=32),

validation\_data=(X\_test, y\_test),

epochs=20)

**7. Evaluation**

* The accuracy curves for training and validation were plotted.

**8. Prediction**

A helper function was implemented to predict on new images:

def predict\_image(img):

img = cv2.resize(img, (IMG\_SIZE, IMG\_SIZE))

img = img.reshape(1, IMG\_SIZE, IMG\_SIZE, 1) / 255.0

pred = model.predict(img)

label = people[int(pred[0][0] > 0.5)]

return label

* Example: When tested on a sample image from the dataset, the model correctly predicted the person’s identity.

**9. Results & Discussion**

* The CNN successfully differentiated between two individuals with high accuracy.
* Data augmentation played a crucial role in improving generalization.
* Overfitting was reduced by applying Dropout layers.
* A limitation is that the model was trained only for two persons; extending it to multiple classes would require modifications (multi-class classification with softmax).

**10. Conclusion**

This assignment demonstrated the implementation of a face recognition model using CNNs with TensorFlow/Keras. The results confirm that CNNs are highly effective in extracting facial features and classifying identities.