# 3. FACE RECOGNITION

|  |  |
| --- | --- |
| **EX.N0 : 3** | **CNN MODEL FOR FACE RECOGNITION** |
| **DATE : 03/03/2025** |

**AIM:**

To build and train a Convolutional Neural Network (CNN) for face recognition using a dataset such as the Labelled Faces in the Wild (LFW) dataset. This model will be capable of recognizing faces from the dataset.

# ALGORITHM:

Step 1: Import necessary libraries.

Step 2: Load and pre-process the LFW dataset.

Step 3: Build the CNN model using Keras Sequential API.

Step 4: Compile the model with appropriate loss function and optimizer. Step 5: Train the model on the training dataset.

Step 6: Evaluate the model on the test dataset.

Step 7: Visualize a few predictions to validate performance.

# PROGRAM:

import tensorflow as tf

from tensorflow.keras import datasets, layers, models from sklearn.model\_selection import train\_test\_split from sklearn import datasets

import numpy as np

lfw\_data = datasets.fetch\_lfw\_people(min\_faces\_per\_person=70, resize=0.4) images = lfw\_data.images # Image data

labels = lfw\_data.target # Labels images = images / 255.0

X\_train, X\_test, y\_train, y\_test = train\_test\_split(images, labels, test\_size=0.3, random\_state=42) model = models.Sequential([

layers.InputLayer(input\_shape=(50, 37, 3)), # LFW image shape layers.Conv2D(32, (3, 3), activation='relu'),

layers.MaxPooling2D((2, 2)),

layers.Conv2D(64, (3, 3), activation='relu'),

layers.MaxPooling2D((2, 2)),

layers.Conv2D(128, (3, 3), activation='relu'), layers.Flatten(),

layers.Dense(128, activation='relu'),

layers.Dense(len(lfw\_data.target\_names), activation='softmax') # Output layer

])

model.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])

model.fit(X\_train, y\_train, epochs=10, batch\_size=32) test\_loss, test\_accuracy = model.evaluate(X\_test, y\_test) print(f"Test Accuracy: {test\_accuracy \* 100:.2f}%") model.save("face\_recognition\_model.h5")

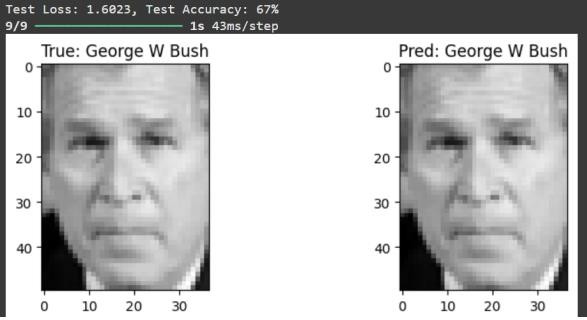
import matplotlib.pyplot as plt predictions = model.predict(X\_test) for i in range(5):

plt.imshow(X\_test[i])

plt.title(f"Predicted: {lfw\_data.target\_names[predictions[i].argmax()]}, Actual:

{lfw\_data.target\_names[y\_test[i]]}") plt.show()

# OUTPUT:

****

**RESULT:**

Thus the Program has been executed successfully and verified.