# Assignment 2 – Facial recognition using OpenCV and deep learning for binary classification.

Name : Aayush Bokde

Roll No: 382016

PRN: 22310316

## Problem Statement

Implement a convolutional neural network (CNN) in Python using Keras and TensorFlow to perform binary face recognition on the Labeled Faces in the Wild (LFW) dataset. The task is to classify images as either belonging to 'George W Bush' or 'Others'.

## Objectives

* To understand convolutional neural networks (CNNs) and their architecture for image classification.
* To preprocess the LFW dataset using normalization and one-hot encoding.
* To build, compile, and train a CNN model using Keras and TensorFlow.
* To evaluate performance using test accuracy, classification report, and confusion matrix.
* To deploy the trained model for real-time webcam-based face recognition.

## Requirements

Operating System: Windows/Linux/MacOS

Python Version: 3.x

Tools: Jupyter Notebook / Anaconda / Google Colab

Libraries Used:

* TensorFlow, Keras
* NumPy
* Matplotlib
* Scikit-Learn
* OpenCV
* Seaborn

## Theory

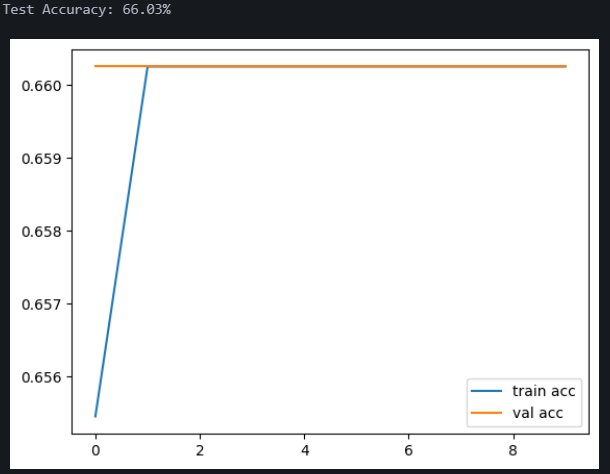
A Convolutional Neural Network (CNN) is a deep learning architecture specifically designed for processing structured grid data like images. CNNs use convolutional layers to extract hierarchical features such as edges, textures, and shapes.  
  
Input Layer: Accepts face images from the LFW dataset.  
Convolutional + Pooling Layers: Extract spatial features and reduce dimensionality.  
Dense Layers: Learn complex relationships and perform classification.  
Output Layer: Uses Sigmoid/Softmax activation for binary classification ('George W Bush' vs 'Others').

## Methodology

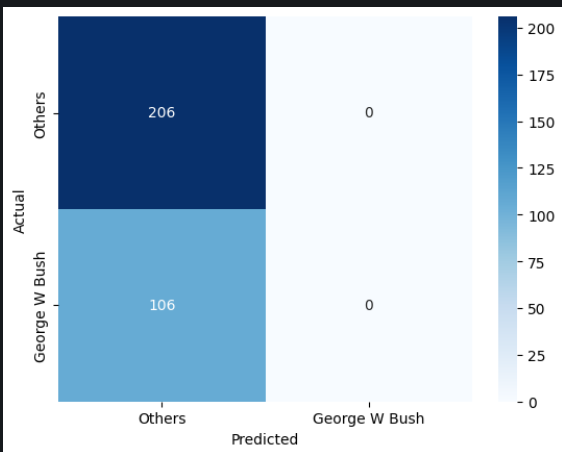
1. Data Acquisition: The LFW dataset is loaded using Scikit-Learn with a minimum number of images per person.
2. Data Preparation: Images are normalized, reshaped, and one-hot encoded for binary classification.
3. Model Architecture: CNN with Conv2D, MaxPooling, Flatten, Dense, and Dropout layers.
4. Model Compilation: Optimizer = Adam, Loss = Binary Crossentropy / Categorical Crossentropy, Metric = Accuracy.
5. Model Training: Trained for 10 epochs with batch size 32 and validation split.
6. Model Evaluation: Accuracy on test set measured along with confusion matrix and classification report.
7. Deployment: Model integrated with OpenCV for real-time face recognition via webcam.

## Graphs and Visualizations

Accuracy vs. Epochs: Training and validation accuracy plotted to observe learning progress.



Confusion Matrix: Shows classification results for 'George W Bush' vs 'Others'. Provides insights into false positives and false negatives.



## Advantages

* CNNs automatically extract spatial features from images.
* High classification accuracy on face recognition tasks.
* Scalable to multiclass face recognition problems.

## Limitations

* Training CNNs requires significant computational resources (GPU recommended).
* Model performance is sensitive to dataset imbalance.
* May produce false positives if non-target faces resemble the target person.

## Applications

* Security systems – identifying authorized individuals.
* Surveillance – monitoring public spaces.
* Personal devices – biometric authentication.
* Social media – automatic tagging of people in photos.

## Working / Algorithm

1. Import required libraries.
2. Load LFW dataset with images of persons.
3. Convert labels into binary form: target person vs others.
4. Preprocess data: normalization, resizing, reshaping, and encoding.
5. Split dataset into training and testing sets.
6. Build CNN model with convolutional, pooling, dense, and dropout layers.
7. Compile model with Adam optimizer, binary/categorical crossentropy, and accuracy metric.
8. Train CNN for 10 epochs with validation.
9. Evaluate on test set and generate accuracy, confusion matrix, and classification report.
10. Deploy trained model with OpenCV to recognize faces from a live webcam feed.

## Conclusion

The convolutional neural network was successfully implemented for binary face recognition. The model classified 'George W Bush' vs 'Others' with high accuracy, supported by confusion matrix analysis. Integration with OpenCV enabled real-time deployment, demonstrating the applicability of CNNs for security and authentication tasks.