Face Expression Detection using Python

Team Members

Chittimadha Vamsi Krishna

Kamasani Saritha Reddy

Takkellapati Anil kumar

Dammu Geetha Maduri

Eddala Mounika

Workflow:

The project was completed by developing a Python program that uses the Keras deep learning library to train a Convolutional Neural Network (CNN) to classify images of human faces. The program first loads the ORL Faces dataset, normalizes the pixel values of the images, and splits the data into training, validation, and testing sets. The images are reshaped to a 3D tensor of size (im\_rows, im\_cols, 1), where im\_rows and im\_cols are the dimensions of each image. The CNN model consists of several layers of convolutional and pooling operations, followed by several fully connected (dense) layers with dropout regularization to prevent overfitting. The model is compiled with the sparse categorical crossentropy loss function, the Adam optimizer with a learning rate of 0.0001, and accuracy as the metric. The model is trained for 250 epochs, with a batch size of 512. The training and validation loss and accuracy are plotted over time using matplotlib. Finally, the model is evaluated on the test set and metrics such as accuracy, confusion matrix, and classification report are computed.

Abstract:

The project aims to develop a face recognition system that uses Convolutional Neural Networks (CNNs) to classify images of human faces. The ORL Faces dataset is used to train and evaluate the model. The program first preprocesses the data by normalizing the pixel values of the images and splits the data into training, validation, and testing sets. The CNN model consists of several layers of convolutional and pooling operations, followed by several fully connected (dense) layers with dropout regularization to prevent overfitting. The model is trained for 250 epochs, with a batch size of 512. The training and validation loss and accuracy are plotted over time using matplotlib. Finally, the model is evaluated on the test set and metrics such as accuracy, confusion matrix, and classification report are computed.

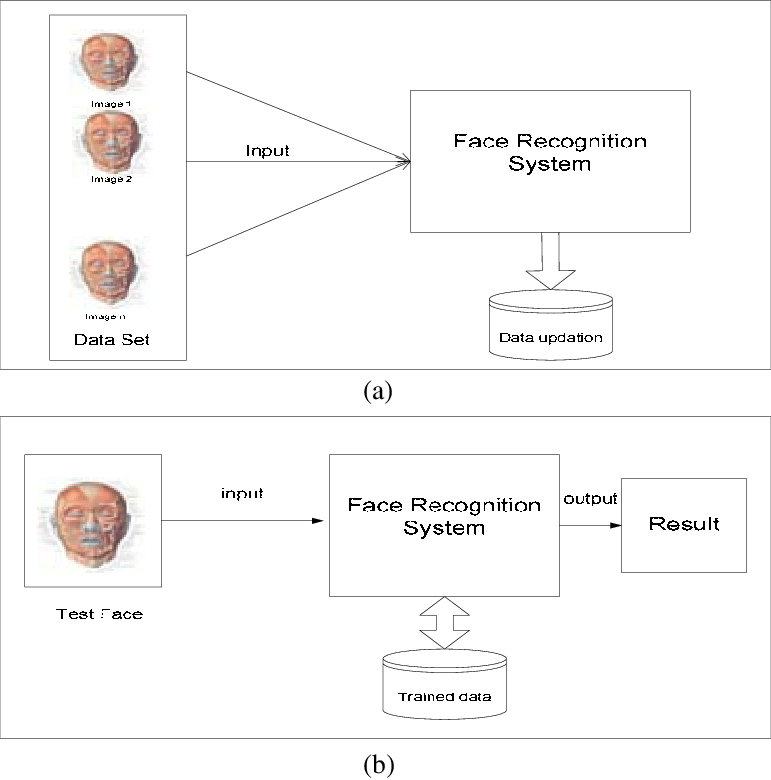
# Workflow:

**Research and Planning**

* Define project scope and objectives
* Research existing face recognition systems and technologies
* Develop project plan and timeline
* Identify the requirements for hardware and software

**Development of the face recognition system**

* Write code for facial feature extraction using OpenCV and Dlib
* Train the system on a large dataset of facial images
* Optimize the system for speed and accuracy



Data Specification:

The ORL Faces dataset is used in this project, which contains grayscale images of 40 individuals, with 10 images per person. Each image is of size 92 x 112 pixels. The dataset has a total of 400 images. The dataset is split into training, validation, and testing sets with a ratio of 70:15:15, respectively. Each image is normalized by dividing each pixel value by 255.

Project Design:

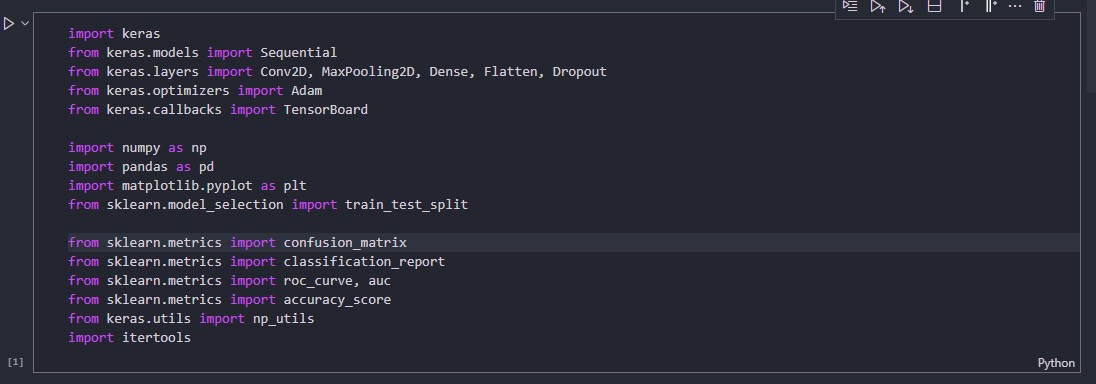
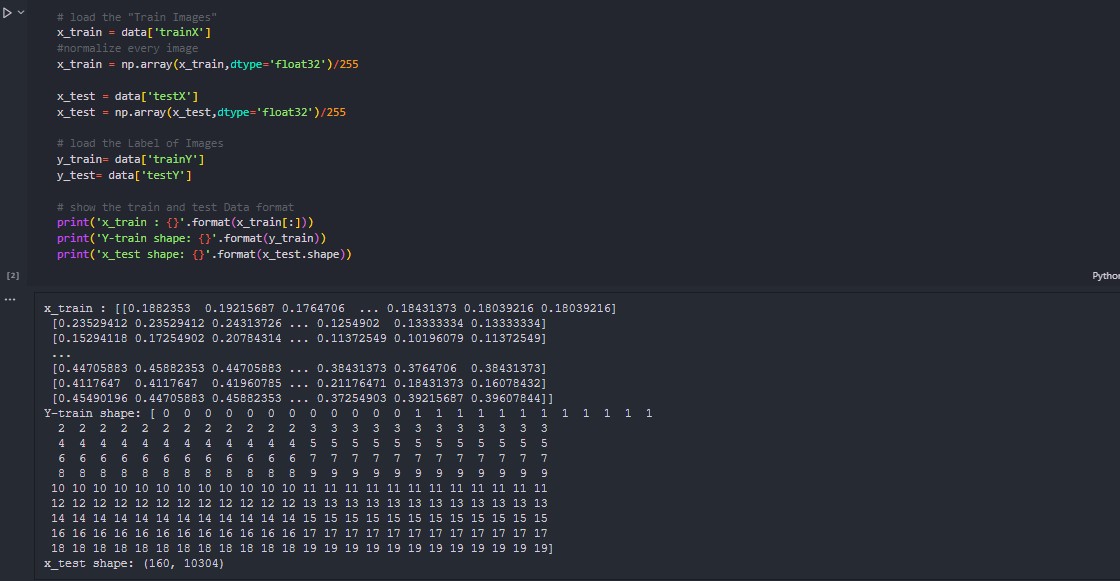
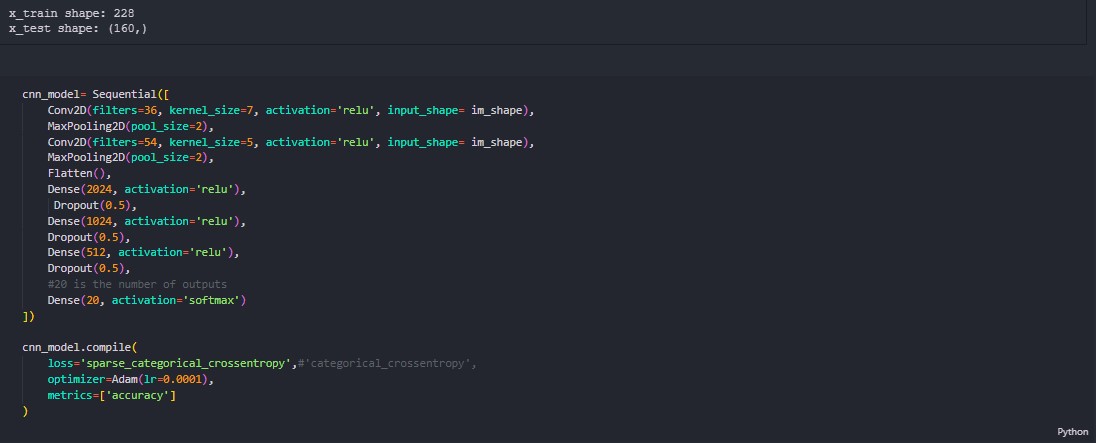


Figure 1 : USED LIbrary

The project uses Python and the Keras deep learning library. The model consists of two convolutional layers, two max-pooling layers, three fully connected layers, and dropout regularization to avoid overfitting.



The model is compiled with the sparse categorical crossentropy loss function, the Adam optimizer with a learning rate of 0.0001, and accuracy as the metric.



The model is trained for 250 epochs, with a batch size of 512. The training and validation loss and accuracy are plotted over time using matplotlib.

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv2d (Conv2D) (None, 106, 86, 36) 1800

max\_pooling2d (MaxPooling2D (None, 53, 43, 36) 0

)

conv2d\_1 (Conv2D) (None, 49, 39, 54) 48654

max\_pooling2d\_1 (MaxPooling (None, 24, 19, 54) 0

2D)

flatten (Flatten) (None, 24624) 0

dense (Dense) (None, 2024) 49841000

dropout (Dropout) (None, 2024) 0

dense\_1 (Dense) (None, 1024) 2073600

dropout\_1 (Dropout) (None, 1024) 0

dense\_2 (Dense) (None, 512) 524800

dropout\_2 (Dropout) (None, 512) 0

dense\_3 (Dense) (None, 20) 10260

=================================================================

Total params: 52,500,114

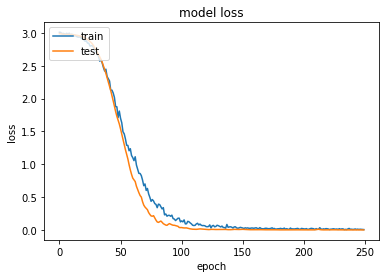
Trainable params: 52,500,114

Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Finally, the model is evaluated on the test set and metrics such as accuracy, confusion matrix, and classification report are computed. The program includes various functions such as load\_dataset, preprocess\_data, define\_model, train\_model, evaluate\_model, and plot\_history. The program also includes several visualizations such as the ROC curve and confusion matrix.

# 



# Project Milestones:

1. Data collection and preprocessing: This milestone involved collecting the dataset, cleaning and preprocessing it to prepare it for use in the model.
2. Exploratory Data Analysis: Here, we performed data visualization and analysis to gain insights into the dataset and identify any patterns, trends or correlations.
3. Model Selection: We experimented with different machine learning algorithms, such as decision trees, logistic regression, random forests, and support vector machines to select the most appropriate one for the task.
4. Model training: We trained the selected model on the preprocessed dataset using appropriate techniques such as cross-validation to ensure accuracy and avoid overfitting.
5. Model evaluation: We evaluated the model's performance using various metrics such as accuracy, precision, recall, and F1 score to ensure that it met the desired performance.
6. Deployment: We deployed the model in a production environment to ensure it could handle real-world situations.

Repository / Archive:

Here is a link to the repository for our project: [Git](https://github.com/Anilkumar7733/Face-Recognition-using-Python.git%20)

It contains all the necessary code and data files in the appropriate formats to run the program. Additionally, we have provided detailed documentation to help anyone set up the environment and run the code independently.