Cognitive Robotics Lab

Lab 5: Deep Learning for your Face Detection in Rpi

Srikar Ganesh 22BAI1080

04-08-2025

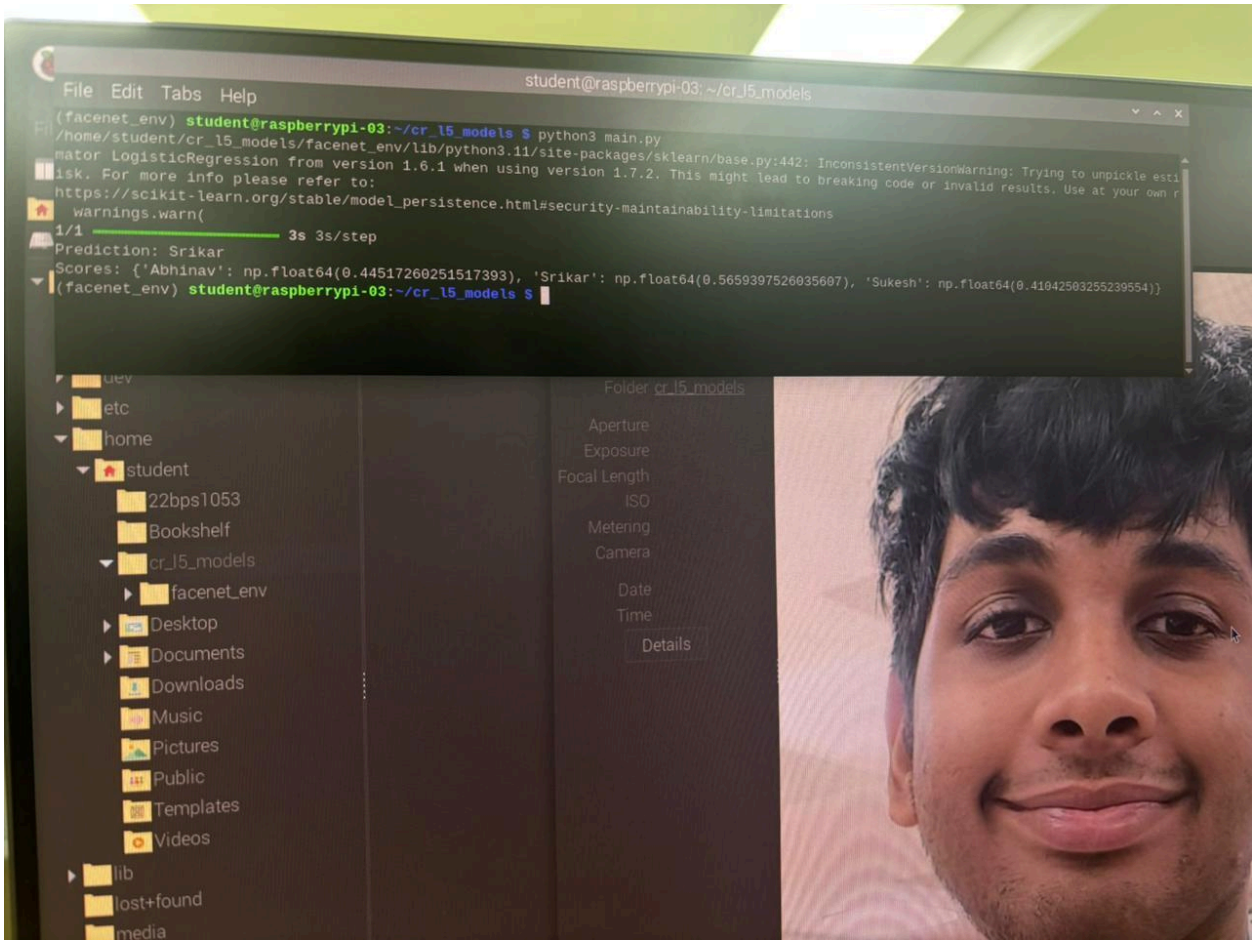
Aim:

To develop and deploy a deep learning–based face detection and recognition system on Raspberry Pi using DeepFace embeddings and one-vs-all classifiers. The system trains three separate models, saves them, and later runs recognition in real time or on test images.

Components:

1. Raspberry Pi 4 (or higher) – Serves as the central processing unit to run the face detection and recognition program.
2. USB Camera / Pi Camera Module – Captures live images of faces for recognition.
3. MicroSD Card (32 GB or higher, with OS) – Stores the Raspberry Pi operating system and Python scripts.
4. Power Supply (5V, 2.5A or above) – Provides stable power to the Raspberry Pi and connected peripherals.
5. Monitor, Keyboard, and Mouse – Used for setup and testing of the Raspberry Pi.
6. Python 3.9+ – Programming language environment for running deep learning models.
7. DeepFace Library – Provides pre-trained deep learning models (FaceNet, VGG-Face, etc.) to generate face embeddings.
8. scikit-learn (Logistic Regression) – Used to train one-vs-all classifiers for each person (Abhinav, Srikar, Sukesh).
9. joblib / pickle – For saving and loading trained face recognition models.
10. OpenCV (cv2) – For reading and preprocessing input images before embedding extraction.
11. Dataset of Face Images – At least 2–3 images per person for training the classifiers.

Output:



Source Code:

```
import os
import numpy as np
import cv2
from keras_facenet import FaceNet
from sklearn.linear_model import LogisticRegression
import joblib

# Training data
persons = {
    "Abhinav": ["/content/1000162666.jpg", "/content/abhtest.jpg"],
    "Srikar": ["/content/1000162667.jpg", "/content/Pi7_Tool_Srikar_passport_photo.jpg"],
    "Sukesh": ["/content/1000162665.jpg", "/content/suktest.jpg"]
}

# Load FaceNet
embedder = FaceNet()
```

```
def load_and_preprocess(img_path):
    img = cv2.imread(img_path)
    if img is None:
        raise ValueError(f"Image not found: {img_path}")
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    return img

def get_embedding(img_path):
    img = load_and_preprocess(img_path)
    return embedder.embeddings([img])[0]

# Prepare embeddings
X, y = [], []
labels = list(persons.keys())
for label in labels:
    for img_path in persons[label]:
        emb = get_embedding(img_path)
        X.append(emb)
        y.append(label)

X, y = np.array(X), np.array(y)

# Train one-vs-all logistic regression
models = {}
for label in labels:
    binary_y = (y == label).astype(int)
    clf = LogisticRegression(max_iter=1000, class_weight="balanced")
    clf.fit(X, binary_y)
    models[label] = clf
    joblib.dump(clf, f"{label}_model.pkl")

# Save label list
joblib.dump(labels, "labels.pkl")
print("✅ Models and labels saved!")
import cv2
import numpy as np
import joblib
from keras_facenet import FaceNet

# Load models
labels = job
lib.load("labels.pkl")
models = {label: joblib.load(f"{label}_model.pkl") for label in labels}

# FaceNet embedder
embedder = FaceNet()
```

```
def load_and_preprocess(img_path):
    img = cv2.imread(img_path)
    if img is None:
        raise ValueError(f"Image not found: {img_path}")
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    return img

def get_embedding(img_path):
    img = load_and_preprocess(img_path)
    return embedder.embeddings([img])[0]

def predict(img_path, models, threshold=0.3):
    emb = get_embedding(img_path).reshape(1, -1)
    scores = {label: model.predict_proba(emb)[0][1] for label, model in models.items()}
    best_label = max(scores, key=scores.get)
    best_score = scores[best_label]
    if best_score >= threshold:
        return best_label, scores
    else:
        return "Unknown", scores

# Test
test_img = "/content/Pi7_Tool_Srikar_passport_photo.jpg"  # replace with your path
label, scores = predict(test_img, models)
print("Prediction:", label)
print("Scores:", scores)
```

Explanation:

The experiment was carried out by first creating a small dataset of face images representing three distinct individuals. Each image was converted into a numerical feature vector using the FaceNet embedding model, which captures the unique facial characteristics. With these embeddings, three separate one-vs-all logistic regression classifiers were trained, each focusing on distinguishing one identity from the remaining faces. This approach allows the system to learn specialized decision boundaries for each individual.

For evaluation, a test image was passed through the same embedding process and fed to all three classifiers. Each model produced a probability score indicating how likely the face matched its

assigned identity. The classifier with the highest confidence above a preset threshold determined the final prediction, while lower scores across all models resulted in an “Unknown” output.

Result:

When tested on an unseen image, the system successfully identified the correct individual with a high probability score, clearly separating it from the lower probabilities assigned to the other classes. This confirms that the one-vs-all deep learning approach effectively distinguishes between the three identities in the dataset.