SOC 2025 Midterm Progress Report

The Eyes of the Machine: A Journey into Face Detection (Project ID: 41)

Patel Subodh Shailendra
Roll No: 24B2509
Department of MEMS, IIT Bombay

June 26, 2025

1. Introduction

This document summarizes my progress in the SOC 2025 project titled *”The Eyes of the Machine: A Journey into Face Detection”*. The goal of this project is to build a Siamese Neural Network-based face detection system using Python and libraries such as TensorFlow and OpenCV. The first phase of this journey has been focused on self-learning — acquiring skills in Python, practicing with data handling libraries, and understanding the theory and architecture of Siamese Networks.

I have structured this report to reflect my journey step-by-step — starting from the basics, building towards implementation, and reflecting on what I've learned.

2. Python Learning Journey

We were encouraged to begin our SOC journey by completing the "100 Days of Code: Python" YouTube playlist by CodeWithHarry. I completed the first 70 lectures, which gave me a strong command over Python syntax and structure.

The playlist covered:

- Basics like variables, loops, functions
- File handling and string formatting
- Object-oriented programming (classes, objects, inheritance)
- Modules, packages, and virtual environments
- Exception handling and file operations

I documented some of my scripts in the GitHub repo under `python_basics/`.

Example - File Writer

```
with open("log.txt", "w") as file:
    file.write("This␣is␣a␣practice␣file␣log␣from␣Python␣basics.")
```

Example - Class and Object

```
class Student:
    def __init__(self, name, branch):
        self.name = name
        self.branch = branch

    def introduce(self):
        print(f"I␣am␣{self.name}␣from␣{self.branch}␣branch.")

s1 = Student("Subodh", "MEMS")
s1.introduce()
```

3. Modules Practice: NumPy, Pandas, Matplotlib

Our mentors shared a document with structured topics and tutorials to learn core Python modules used in machine learning workflows.

3.1 NumPy - Numerical Python

NumPy is essential for scientific computing. I practiced:

- Creating arrays and reshaping
- Matrix multiplication and dot product
- Broadcasting and slicing
- Aggregation functions: mean, std, sum, max, etc.

Sample - Broadcasting in NumPy

```
import numpy as np
a = np.array([1, 2, 3])
b = np.array([[10], [20], [30]])
print("Broadcasted␣Result:\n", a + b)
```

3.2 Pandas - Data Analysis Tool

Pandas helped me understand structured data. I worked with:

- Series and DataFrames
- Reading CSVs and Excel files
- Indexing and filtering rows

- GroupBy and value$_counts$()

Sample - Filtering a DataFrame

```
import pandas as pd
df = pd.DataFrame({’Name’: [’A’, ’B’], ’Age’: [21, 19]})
print(df[df[’Age’] > 20])
```

3.3 Matplotlib - Visualization

Matplotlib allowed me to convert data into visuals.

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4]
y = [5, 8, 6, 9]

plt.plot(x, y, marker=’o’, linestyle=’-’, color=’green’)
plt.title("Performance␣Chart")
plt.xlabel("Days")
plt.ylabel("Score")
plt.grid(True)
plt.show()
```

These scripts are documented in the modules_practice/ folder.

4. Siamese Neural Network: Architecture and Concept

After learning Python, we were introduced to the architecture that we’ll be implementing in this project: the Siamese Neural Network.

What is it?

A Siamese Network consists of two identical neural networks joined at their outputs. It is used to determine the similarity between two inputs.

Use Case: Given two face images, decide whether they belong to the same person.

Base Network Code (Keras)

```
from tensorflow.keras import layers, Model, Input

def base_model(input_shape):
    input = Input(shape=input_shape)
    x = layers.Conv2D(64, (3,3), activation=’relu’)(input)
    x = layers.MaxPooling2D()(x)
    x = layers.Conv2D(128, (3,3), activation=’relu’)(x)
    x = layers.MaxPooling2D()(x)
    x = layers.Flatten()(x)
    x = layers.Dense(128, activation=’relu’)(x)
    return Model(input, x)
```

I have saved this script as siamese_architecture.py in the siamese_demo/ folder.

Testing Similarity

```
import numpy as np

img1 = np.random.rand(1, 100, 100, 1)
img2 = np.random.rand(1, 100, 100, 1)

e1 = model.predict(img1)
e2 = model.predict(img2)

distance = np.linalg.norm(e1 - e2)
print("Similarity:", distance)
```

5. Repository Structure and Resources

My GitHub repository is structured to match the phases of my learning and coding practice.

```
python_basics/       → Code from CodeWithHarry playlist
modules_practice/    → Notebooks for NumPy, Pandas, Matplotlib
siamese_demo/        → Siamese architecture and tests
resources/           → Mentor PDFs and learning links
README.md            → Project summary and links
progress_report.pdf → This report
```

In the `resources/` folder, I have included:

- **DOC-20250613-WA0002.pdf** — Learning Modules file
- **Siamese Neural Network.pdf** — Model explanation
- **learning_links.txt** — All YouTube/doc links used
- **notes_summary.txt** — My personal takeaways

6. Key Learnings and Reflections

- Python is beginner-friendly but deep — a solid language for ML.
- NumPy taught me how to think in terms of matrices and tensors.
- Pandas helped me analyze, filter, and prepare data quickly.
- Matplotlib is extremely helpful when debugging with visuals.
- OpenCV is powerful for working with images in real-time.
- Siamese Networks are ideal for low-data image comparison tasks.

I also understood that one of the biggest strengths of ML is reproducibility. I now comment my code more and document what each cell is doing.

7. Next Steps

1. Prepare image datasets and form positive/negative pairs
2. Use contrastive loss and triplet loss for training
3. Run training and validation phases with callbacks
4. Build a simple face-matching GUI using OpenCV

8. Final Words

This report captures my first-phase learning of the SOC 2025 journey. It's been a great learning experience, and I'm now confident to step into the implementation stage of the project.