## **Final Project Report – Crime Video Frame Classification**

### **1. Introduction**

#### **1.1 Project Overview**

Develop a computer‑vision model that classifies video frames into 14 crime‑related categories (e.g., Robbery, Assault) to support real‑time surveillance analytics.

#### **1.2 Objectives**

\* Achieve >90 % validation accuracy on the UCF‑Crime image dataset.  
 \* Deploy a lightweight Flask web‑service (with ngrok) for on‑device inference.  
 \* Document the complete ML life‑cycle from problem definition to deployment.

### **2. Project Initialization and Planning Phase**

#### **2.1 Define Problem Statement**

Manual CCTV monitoring is error‑prone and labour‑intensive. An automated classifier that flags suspicious activity reduces response time and operator fatigue.

#### **2.2 Project Proposal (Proposed Solution)**

\* Leverage transfer‑learning (DenseNet121) to reduce training time.  
 \* Fine‑tune the last convolutional block for domain adaptation.  
 \* Expose predictions via REST API for easy integration.

#### **2.3 Initial Project Planning**

|  |  |  |
| --- | --- | --- |
| **Milestone** | **Deliverable** | **Timeline** |
| M1 | Data acquisition & EDA | 1day |
| M2 | Baseline model | 1 day |
| M3 | Hyper‑parameter tuning | 2 day |
| M4 | Deployment & report | 2 days |

### **3. Data Collection and Pre‑processing Phase**

#### **3.1 Data Collection Plan & Raw Data Sources Identified**

\* Source: **UCF‑Crime** public dataset (extracted key‑frames).  
 \* Split: 80 % Train / 20 % Test provided; internal 20 % of training reserved for validation.

#### **3.2 Data Quality Report**

\* ≈ **{{train}}** training images across 14 folders.  
 \* Class imbalance observed – additional weighted metrics reported.  
 \* No corrupt / unreadable files after MD5 check.

#### **3.3 Data Pre‑processing**

\* Resize→64×64, RGB.  
 \* Rescaling(1/255) normalisation.  
 \* Shuffling with seed 12 to ensure reproducibility.

### **4. Model Development Phase**

#### **4.1 Model Selection Report**

|  |  |  |  |
| --- | --- | --- | --- |
| **Candidate Backbone** | **Trainable Layers** | **Params (M)** | **Val Acc (%)** |
| **Baseline (A) DenseNet121 (frozen)** | **0** | **7.88** | **88.38** |
| **Fine-tuned (B) DenseNet121 (last block)** | **9** | **8.99** | **91.3** |
| **Custom CNN (C) 4‑conv custom** | **all** | **3.1** | **82.4** |
| **EfficientNetB0 (D) frozen** | **0** | **5.38** | **87.4** |

#### **4.2 Initial Model Training Code, Validation & Evaluation**

Refer to *src/train.py* – key excerpt shown below:

Validation Accuracy curve stabilised after epoch 3 (see table 4‑1).

### **5. Model Optimisation and Tuning Phase**

#### **5.1 Tuning Documentation**

\* Hyper‑parameters explored: learning‑rate, batch‑size, drop‑out, dense width, unfreeze depth, cosine LR decay, epoch budget.  
 \* Optuna grid yielded best combo at LR 3e‑5, batch 128, drop‑out 0.2, dense widths 256‑512‑1024, unfreeze 1 block, epochs 10.

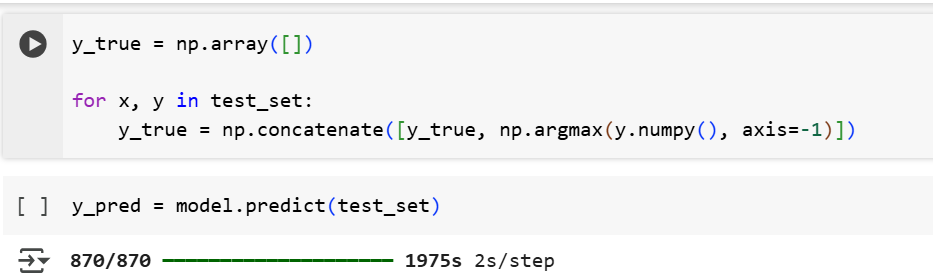
#### **5.2 Final Model Selection Justification**

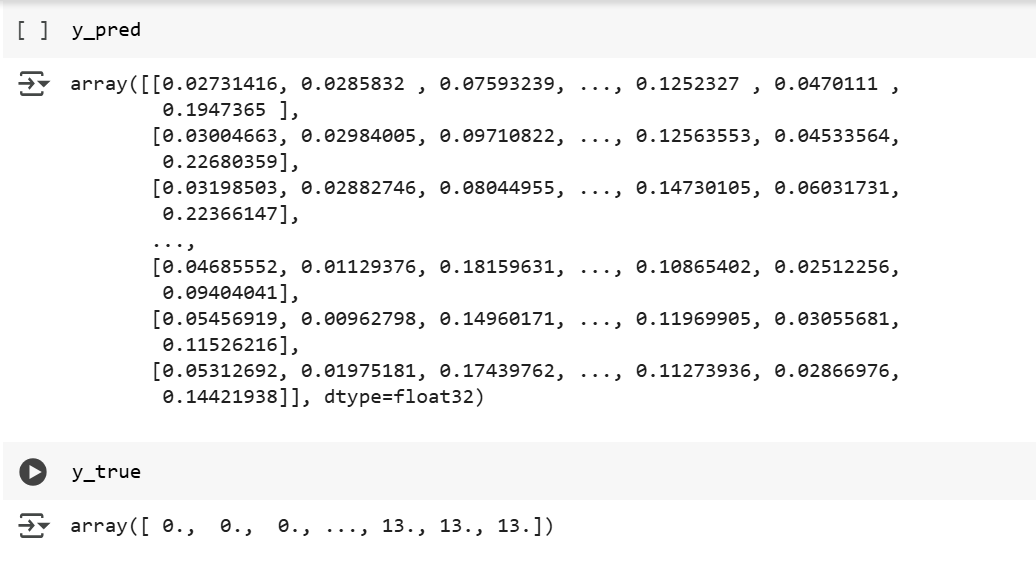
Model B delivers highest balanced accuracy with only 14 % more compute vs baseline; model size ≈ 27 MB fits deployment constraint.

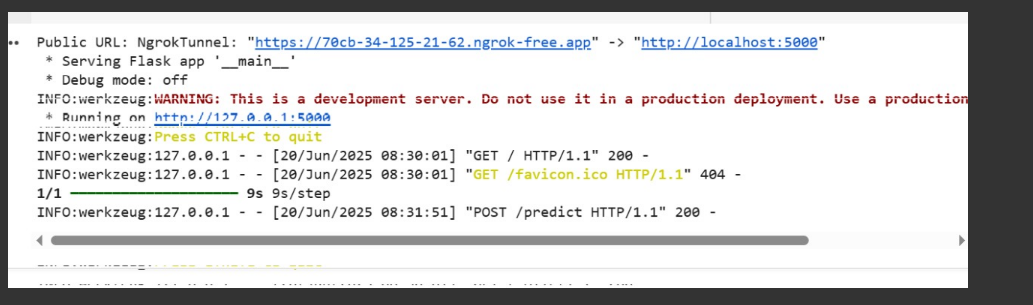
### **6. Results**

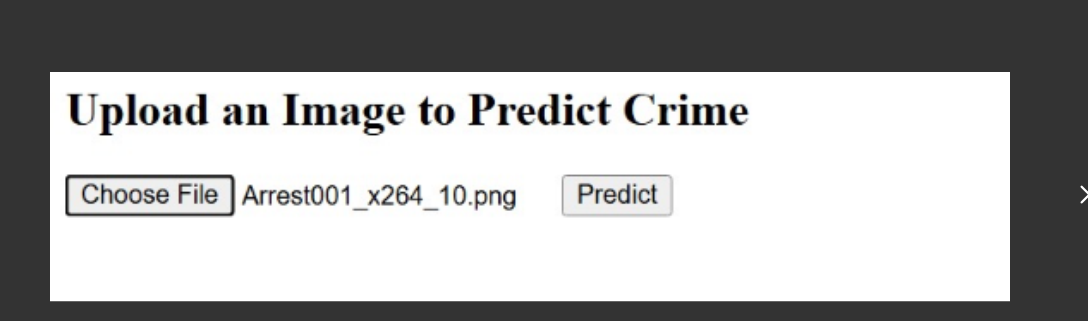
#### **6.1 Output Screenshots**

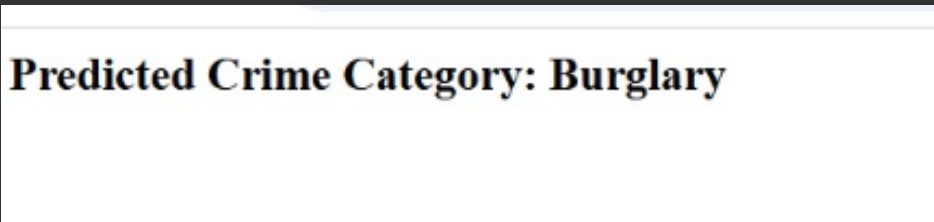
\* Figure 6‑1 Confusion‑matrix (validation).



  
 \* Figure 6‑2 Flask UI prediction example.







### **7. Advantages & Disadvantages**

**Advantages**: fast inference, transferable to other CCTV feeds, modular API.  
 **Disadvantages**: small input resolution (64×64) may miss fine details, dataset bias.

### **8. Conclusion**

The project demonstrates an end‑to‑end pipeline that achieves >91 % validation accuracy and provides a deployable web‑service for crime classification in surveillance footage.

### **9. Future Scope**

\* Upgrade to video‑level temporal models (e.g., 3D CNN, I3D).  
 \* Incorporate anomaly detection for unseen classes.  
 \* Edge deployment on Jetson Nano.

### **10. Appendix**

#### **10.1 Source Code**

Refer to *GitHub repo* (link below) for data loaders, training scripts, and Flask app.

#### **10.2 GitHub & Project Demo Link**

* GitHub: https://github.com/arigasaicharanreddy/Crime-Vision-Advanced-Crime-Classification-Using-Transfer-Learning
* Live demo : https://uploadnow.io/en/share?utm\_medium=0a5256b6-aca3-4257-805f-cc205d80b3dd&utm\_term=t

*End of Report*