EyeGuard: A Computer Vision-Based Approach to Detecting Shoplifting in Retail Stores – Project Progress Report

1. Project Concept & Planning

- Chose **EyedGuard**, a shoplifting detection system.
- Defined the objective: **Detect shoplifting behavior using AI & computer vision**.
- Conducted research on **existing methods and datasets** for theft detection.

2. Dataset Collection & Preprocessing

- Collected **video datasets** related to shoplifting activities.
- Extracted frames, resized images, and applied **data augmentation**.
- Labeled data into **shoplifting vs. normal behavior** categories.

3. Model Selection & Training

- To achieve robust shoplifting detection, multiple deep learning architectures were implemented and tested:
    1. **ConvLSTM (Convolutional LSTM)** – Used in the first approach to capture both **spatial (image details) and temporal (movement patterns) features** from surveillance videos.
    2. **Frame-Level CNN** – The second approach used a **Convolutional Neural Network (CNN)** to process frames individually, treating the problem as an image classification task.
    3. **MovieNet Model** – In the third approach, **MovieNet** was experimented with to analyze human actions and behaviors over a sequence of frames.
    4. **CNN-RNN Hybrid** – The fourth approach combined a **CNN for spatial feature extraction** with an **RNN (LSTM/GRU) for sequential pattern analysis**, improving motion-based detection.
    5. **YOLO + ResNet for Bounding Box Detection** – The fifth approach incorporated **YOLO (You Only Look Once)** for **real-time object detection**, combined with **ResNet** for further classification refinement, focusing on detecting **suspicious human behavior within bounding boxes**.

- **Code Availability:** The complete implementation of these models is **physically available on GitHub** for reference and future improvements.

- Trained the model and evaluated performance using **accuracy, precision, recall, and F1-score**.

4. Real-Time Detection Implementation

- Integrated **OpenCV and Deep Learning** for real-time video analysis.
- Used **cv2.dnn module** to load the trained model and detect suspicious activities.
- Successfully tested real-time detection on sample videos.

Next Steps – To Be Continued

5. Out-of-Distribution (OOD) Detection *(Next Approach)*

- Implement **OOD detection techniques** (Mahalanobis distance, OpenMax) to identify unknown activities.

6. Deployment & API Integration *(Next Approach)*

- Develop an API (Flask/FastAPI) to handle **video input and model inference**.
- Connect the system to a **dashboard or mobile app** for shop owners.

7. Testing & Performance Optimization *(Next Approach)*

- Test the system in **different environments (lighting, angles, occlusions)**.
- Improve model performance with **hyperparameter tuning** and additional training data.

8. Final Presentation & Report Preparation *(Next Approach)*

- Create **detailed documentation and analysis** of results.
- Develop a **presentation/demo** showcasing real-time detection.
- Compile findings and discuss **future improvements**.