Real-Time Crowd Monitoring Using CSRNet

Kavya mehta

1. Introduction

Milestone 3 focuses on integrating the trained CSRNet model into a real-time crowd monitoring system.
The goal is to process continuous video streams (e.g., webcam or video file), apply the CSRNet deep learning model on each frame,
and display real-time crowd estimations using visualization tools such as Gradio.

2. Objective

The primary objective of this milestone is to design, develop, and demonstrate a real-time pipeline for crowd monitoring using the trained CSRNet model.
This simulation replicates how CCTV systems perform crowd analysis for safety management, congestion detection, and automated alert systems.

3. Components of the Real-Time Pipeline

3.1 Capture Frames from Video/Webcam

Instead of relying on an actual CCTV camera, the laptop webcam or any video file can be used as a continuous video source.
OpenCV (cv2) is used for reading frames either from a webcam (cv2.VideoCapture(0)) or video file.

3.2 Preprocessing Each Frame

Each frame must be preprocessed in a similar manner to the preprocessing used during training:
• Resize the frame (optional but recommended for faster inference).
• Convert BGR → RGB.
• Normalize using ImageNet mean and std.
• Convert to a PyTorch tensor.
• Add batch dimension.
Consistency between training preprocessing and inference preprocessing is crucial.

3.3 Running CSRNet on Each Frame

Once the frame is preprocessed, it is passed into the trained CSRNet model.
CSRNet outputs a density map, where the sum of all pixel values represents the estimated

number of people.

3.4 Displaying Real-Time Crowd Counts using Gradio

Gradio allows building a clean and interactive web application that:
• Displays the live webcam feed.
• Runs the model in real-time on each frame.
• Shows the estimated crowd count.
• Optionally displays the density heatmap.

3.5 Overlay Optional Density Heatmaps

The output density map can be normalized and resized to match the original frame, and then blended with the original image to visualize areas with high crowd concentration. This helps in understanding model behavior and identifying potential congested areas.

3.6 Trigger Automated Alerts

Alerts can be generated when the predicted count exceeds a user-defined threshold.
Examples:
• Display an ALERT message in the UI.
• Highlight the frame with a red border.
• Log the alert event.
Such alerts mimic real-world systems that notify security teams about overcrowded regions.

4. System Workflow

1. Start Gradio interface.
2. Webcam/video stream begins.
3. For each incoming frame:
   a. Preprocess frame.
   b. Predict density map using CSRNet.
   c. Compute crowd count (sum of density).
   d. Show output frame + count.
   e. Trigger alerts if threshold exceeded.
4. System runs continuously until stopped.

5. Applications

- Public Area Monitoring (stations, malls, festivals)
- Queue and Queue Density Monitoring
- Emergency Evacuation Management
- Real-time Crowd Analytics Dashboards
- Smart Surveillance Systems

6. Conclusion

Milestone 3 demonstrates the practical usability of CSRNet beyond offline dataset training. Integrating it with a real-time stream and visualization interface helps bridge theory and deployment.
This milestone is essential for understanding how AI-based crowd counting can be used in real-world surveillance and monitoring systems.