Crowd Surveillance using YOLOv8 and BoT-SORT Tracking

Akshat Jha (B23336)
Harsh Vardhan Sharma (B23373)

December 09, 2025

Contents

Abstract

This project presents a crowd surveillance system utilizing YOLOv8 for pedestrian detection and BoT-SORT for tracking in dense environments. We fine-tuned YOLOv8 on the MOT20 dataset to specialize in person detection, addressing challenges like occlusion and high density. The system achieves high accuracy with mAP@50 of 0.982 on validation sequences. We discuss the methodology, results on benchmark and real-world videos, limitations, and future enhancements for robust multi-object tracking in surveillance applications. The transition from head detection to full-body detection improved tracking stability and occlusion robustness. This work contributes to automated monitoring in public spaces, enhancing safety and analytics.

1 Introduction

Crowd surveillance has become an essential component of modern urban management and public safety systems. With the rapid urbanization and increasing population density in cities worldwide, public spaces such as stadiums, airports, shopping malls, and transportation hubs are frequently overcrowded. These environments pose significant risks, including stampedes, terrorist activities, and public health concerns, as evidenced by historical incidents like the Love Parade disaster in 2010 or the Itaewon crowd crush in 2022. Automated crowd surveillance systems leverage computer vision techniques to monitor, analyze, and predict crowd behaviors in real-time, enabling proactive interventions. Traditional surveillance methods rely heavily on human operators monitoring CCTV feeds, which is not only labor-intensive but also prone to errors due to fatigue and limited attention spans. Manual approaches fail to scale in large-scale deployments, where hundreds of cameras may be involved. In contrast, automated systems powered by artificial intelligence can process vast amounts of video data efficiently, providing insights into crowd density, flow patterns, and anomalous behaviors. The applications of crowd surveillance extend beyond security. In robotics and autonomous navigation, understanding crowd dynamics allows robots to navigate safely through human-populated areas. In transportation, crowd-flow analytics can optimize traffic management and public transit schedules. Behavioral analysis can aid in marketing strategies for retail spaces or event planning. However, developing effective crowd surveillance systems faces several technical challenges. Heavy occlusion occurs when individuals are partially or fully blocked by others, making detection difficult. High crowd density results in small target sizes relative to the frame, complicating accurate bounding box predictions. Real-time inference is crucial for practical deployment, requiring models that balance accuracy and computational efficiency. This project builds upon our midterm progress, where we initially explored head detection using the YOLOv8n model trained on the JHU-Crowd++ dataset. While head detection seemed promising for counting in dense crowds, it revealed limitations in tracking applications. Head bounding boxes are inherently small, leading to low Intersection-over-Union (IoU) values between consecutive frames, which disrupts motion-based trackers like BoT-SORT. Additionally, heads are easily occluded, whereas full-body detection provides more visible features (e.g., torso, limbs) for robustness. Full-body boxes also offer better contextual intuition for surveillance operators. Consequently, we pivoted to full-body pedestrian detection using the MOT20 dataset, a benchmark known for its challenging dense scenes. Our objective is to develop an end-to-end pipeline integrating fine-tuned YOLOv8 for detection and BoT-SORT for tracking, specialized for crowd surveillance. This system aims to achieve high accuracy in detection and stable tracking, even under occlusion and density variations. The remainder of this report is organized as follows: Section 2 reviews relevant literature, Section 3 details the methodology, Section 4 presents results and discussion, Section 5 provides demo and testing instructions, Section 6 concludes the work, and Section 7 outlines future scope.

2 Literature Review

The field of crowd surveillance has evolved significantly with advancements in deep learning, particularly in object detection and multi-object tracking (MOT). This section reviews key works on detection models like YOLO, tracking algorithms such as BoT-SORT, benchmark datasets like MOT20, and overall reviews on deep learning applications in crowd analysis.

2.1 Object Detection in Crowds

Object detection forms the foundation of crowd surveillance systems. The YOLO (You Only Look Once) series has gained prominence for its real-time capabilities. Redmon et al. (2016) introduced YOLO as a single-stage detector that predicts bounding boxes and class probabilities directly from full images, achieving high speed at the cost of some accuracy compared to two-stage models like Faster R-CNN. Subsequent versions improved upon this. YOLOv3 incorporated multi-scale predictions and better feature extractors. YOLOv4 and YOLOv5 focused on optimization for deployment. YOLOv8, by Ultralytics, introduces anchor-free detection, mosaic augmentation, and better backbone networks, making it suitable for dense crowd scenarios. Recent studies have adapted YOLO variants for crowd-specific challenges. For instance, Gündüz and Işık (2023) proposed a YOLO-based method for real-time crowd detection from video sequences, focusing on measuring crowd size in predefined regions [1]. Their approach demonstrated effective performance in dynamic environments by integrating spatial and temporal features. In a comparative analysis, researchers evaluated YOLOv3, YOLOv5, YOLOv7, and YOLOv8 for crowd analysis, highlighting improvements in detection accuracy and speed across versions [2]. The study showed that YOLOv8 achieves the best balance for real-time applications in surveillance. Ahmed et al. (2023) utilized YOLOv5 with feature enhancements for human body detection and counting in crowded scenes, achieving high precision in dense scenarios [3]. They incorporated attention mechanisms to focus on relevant regions, reducing false positives. For sparse regions, YOLO models were evaluated, with YOLOv7 and YOLOv11 showing superior performance in real-world crowd detection [4]. The evaluation included metrics like mAP and inference time, emphasizing deployment feasibility. Dense crowd detection has been addressed by improving YOLOv7 with lightweight attention mechanisms to reduce missed detections at image edges [5]. This modification improved recall in high-density areas. Low-light and dense crowd scenarios were tackled by an improved YOLO network, outperforming baselines like YOLOX-s in pedestrian average precision [6]. The enhancements included better handling of illumination variations. Comprehensive reviews, such as Elharrouss et al. (2024), survey deep learning techniques for object detection and crowd analysis, emphasizing end-to-end models [7]. They discuss the shift from traditional methods to CNN-based detectors. Additional works include crowd detection using YOLOv8 [8] and deep learning-based density estimation [9], which complement detection with counting tasks.

2.2 Multi-Object Tracking

Tracking builds upon detection by associating objects across frames. The SORT algorithm by Bewley et al. (2016) uses Kalman filters for motion prediction and the Hungarian algorithm for data association, providing a simple yet effective baseline. BoT-SORT extends this by incorporating camera motion compensation and robust re-identification. Aharon et al. (2022) introduced BoT-SORT as a robust MOT tracker that combines motion and appearance cues

with camera-motion compensation, achieving state-of-the-art results on benchmarks [10]. It handles occlusions better by using re-identification features. An improved version fused millimeter-wave radar and camera features for robotic applications [11], enhancing tracking in adverse conditions. Sports-specific adaptations, like Vol-Bot-SORT for volleyball player tracking, integrate YOLOv8 detections with BoT-SORT for effective multi-target tracking [12], showing versatility.

2.3 Benchmarks and Datasets

The MOT20 dataset is a cornerstone for evaluating MOT in crowded scenes. Dendorfer et al. (2020) presented MOT20 with eight sequences of dense pedestrian videos, emphasizing occlusion and scale variations [13]. It has been used to benchmark trackers, including FairMOT variants [14].

2.4 Reviews on Deep Learning in Crowd Surveillance

Several reviews synthesize the field. Elharrouss et al. (2024) provide a comprehensive overview of deep learning for object detection and crowd analysis [7]. Tripathi et al. (2019) review intelligent video surveillance using deep learning for crowd analysis [17]. Crowd behavior analysis through deep learning is revisited by Bendali-Braham et al. (2020), focusing on anomaly detection [15]. Recent trends in crowd management using DL highlight supervised and unsupervised approaches [19]. Abnormal behavior detection reviews cover traditional and DL methods [18]. Other works include deep CNN for crowd density estimation [20] and brief reviews on crowd behavior analysis [16]. These works collectively inform our choice of YOLOv8 and BoT-SORT for the project, leveraging their strengths in dense, real-time scenarios.

3 Methodology

Our methodology encompasses dataset preparation, model training, and the full surveillance pipeline. We emphasize transfer learning and data augmentation for robustness.

3.1 Dataset and Preprocessing

The MOT20 dataset was selected for its challenging nature, featuring dense pedestrian scenes with occlusion and clutter [13]. It comprises four training sequences (MOT20-01, MOT20-02, MOT20-03, MOT20-05), one validation sequence (MOT20-04), and three testing sequences (MOT20-06, MOT20-07, MOT20-08). Each sequence is high-resolution video with ground-truth annotations for pedestrians. Annotations in MOTChallenge format (gt.txt) include frame number, ID, bounding box (x, y, w, h), confidence, and other attributes. We developed a custom Python script to convert them to YOLO format. The process involved:

1. Parsing gt.txt to extract bounding boxes per frame.
2. Filtering detections with confidence below 0.5 to ensure quality.
3. Normalizing coordinates to [0,1] range: center x = (x + w/2)/image_width, similarly for y, w, h.
4. Generating one .txt file per frame with lines as "class x y w h" (class=0 for person).

This resulted in a structured dataset: train/valid splits with images and labels folders, compatible with Ultralytics YOLO training API.

3.2 Training Pipeline

We fine-tuned YOLOv8s, pretrained on COCO, for single-class "person" detection. Transfer learning leverages pre-learned features for faster convergence and better performance on limited data. The YOLOv8 architecture includes a CSPDarknet backbone, PANet neck, and anchor-free head. Fine-tuning involved freezing early layers and training the head on MOT20.
Configuration details:

Parameter	Value
Image Size	640 × 640
Batch Size	4
Epochs	50
Optimizer	AdamW
Learning Rate	0.002
Momentum	0.9
Device	NVIDIA Tesla T4

Table 1: Training Hyperparameters

Training was conducted on Google Colab with GPU acceleration. The loss function is a combination of box loss (CIoU), classification loss (BCE), and distribution focal loss for better regression.
Data augmentation was crucial for robustness:

1. Blur and MedianBlur: To handle motion blur in video frames.

2. ToGray: For grayscale robustness in low-color CCTV.

3. CLAHE (Contrast Limited Adaptive Histogram Equalization): Contrast enhancement for varying lighting.

These were applied probabilistically during training via Ultralytics. The trained model was exported as best.pt (PyTorch) and best.onnx for inference flexibility.

3.3 Testing and Analytics Pipeline

The end-to-end system processes input videos as follows:

1. Frame extraction using OpenCV to save individual images.

2. YOLO inference on frames to get detections (bounding boxes, confidences).

3. Convert YOLO outputs to MOT format: frame, ID (temp), box, conf, etc.

4. Apply BoT-SORT: Uses Kalman filter for prediction, IoU for association, handles occlusions with re-association.

5. Annotate frames with boxes and IDs, compile into output video.

The pipeline is implemented in a Jupyter notebook for reproducibility, allowing one-click execution on new videos. For testing and advanced analysis, the system includes an extended pipeline that processes novel test videos (e.g., CCTV or drone footage) and generates actionable insights:

1. **Input**: Injection of novel test video (CCTV/Drone footage).

2. **Core Inference**: YOLOv8 Detection → BoT-SORT Tracking.

3. **Temporal Analytics**: Frame-by-frame occupancy counting and dwell time estimation.

4. **Spatial Analytics**: Density heatmaps, velocity flow fields, and trajectory mapping.

5. **Event Logic**: Zone entry/exit monitoring and social distancing violation checks.

6. **Output**: Annotated video render, CSV statistical logs, and visualization plots.

The pipeline processes raw tracking data into actionable insights (Heatmaps, Dwell Times, Flow) automatically.

4 Results and Discussion

Validation on MOT20-04 (held-out during training) at epoch 50 yielded impressive metrics, indicating the model's effectiveness: High precision and recall show low

Metric	Value
mAP@50	0.982
mAP@50-95	0.837
Precision	0.986
Recall	0.956
Fitness	0.837

Table 2: Validation Metrics on MOT20-04

false positives/negatives, while mAP confirms robust detection across IoU thresholds. Qualitative results on MOT20 test sequences demonstrate accurate detection in highly crowded scenes, with bounding boxes tightly fitting pedestrians even under partial occlusion. Tracking maintains good ID continuity for moderate movements and occlusions, but ID switches occur in extreme overlaps where objects are fully hidden for several frames. On real-world videos, including indoor corridors (e.g., university halls), outdoor footpaths (e.g., busy streets), and CCTV-style footage (low resolution, varying angles), the model generalized well. Detection remained accurate despite domain shift from MOT20's specific camera views. Tracking stability was better in less dense real-world scenarios. The testing and analysis pipeline was applied to these videos, generating density heatmaps and trajectory maps that highlighted crowd flow patterns and potential bottlenecks. Temporal analytics provided insights into average dwell times and occupancy trends, useful for behavioral analysis. Discussion: The pivot from head to body detection significantly improved IoU-based tracking, as larger boxes provide more stable overlaps. Occlusion robustness increased due to more visible body parts. However, the current BoT-SORT implementation relies primarily on motion and spatial information, lacking deep appearance features, which leads to ID switches in severe occlusions. Quantitative MOT metrics like MOTA (Multi-Object Tracking Accuracy) and IDF1 (ID F1 score) were not computed yet but would provide deeper insights. Limitations include struggles in extremely dense, static crowds where motion cues are minimal, and potential overfitting to MOT20's style despite augmentations. Overall, the results validate the system's suitability for crowd surveillance, with moderate detection accuracy enabling reliable tracking and analytics in challenging environments.

Figure 1: Example Detection and Tracking in a Dense Crowd Scene from MOT20

5 Demo and Testing Instructions

To demonstrate and test the crowd surveillance system, we provide a Jupyter notebook and associated files in a shared Google Drive folder. The folder contains the fine-tuned models, results, and demo scripts. The Drive link is: Google Drive Folder. Files in the folder include:

- **results**: Subfolder containing output examples.
- **best.pt**: Fine-tuned YOLOv8 model in PyTorch format.
- **best.onnx**: Fine-tuned model in ONNX format for deployment.
- **mot20_iteration1.ipynb**: Notebook for MOT20 data processing.
- **robot_vision_crowd_surveillance_demo_1.ipynb**: Main demo notebook for running the pipeline.

To test the system: (change paths for

1. Download the `robot_vision_crowd_surveillance_demo_1.ipynb` and required models (`best.pt` or `best.onnx`) from the Drive folder.
2. Open the notebook in Google Colab or a local Jupyter environment.
3. Execute all cells in sequence. The pipeline will run detection and tracking on the uploaded video.
4. In cell 4, upload your own video file for processing.
5. After completion, download the output video from the path `/content/robo_pipeline/output_videos`.
6. View the annotated output video locally to observe the detection and tracking results.

This setup allows users to test the system on custom videos, verifying its performance in various scenarios.

6 Conclusions

We successfully implemented a crowd surveillance system integrating YOLOv8 for pedestrian detection and BoT-SORT for tracking. Key achievements include: - Custom conversion of MOT20 annotations to YOLO format. - Fine-tuning YOLOv8s via transfer learning, achieving 0.982 mAP@50 on validation. - Development of an end-to-end pipeline for video processing and annotation. - Demonstration of effectiveness on benchmark datasets and real-world videos. The system addresses core challenges in crowd monitoring, such as occlusion and density, and contributes to applications in public safety, robotics, and analytics. By specializing the detector on dense scenes, we ensured high reliability, making it a practical tool for automated surveillance.

7 Future Scope

To enhance the system, future work could include:

- Adopting instance segmentation models like Mask R-CNN for better occlusion handling through pixel-level masks.
- Integrating DeepSORT or ByteTrack with appearance embeddings (e.g., ReID features from OSNet) to reduce ID switches in long occlusions.
- Computing standard MOT metrics (MOTA, IDF1, HOTA) for quantitative tracking evaluation.
- Adding high-level analytics: Real-time crowd density heatmaps, flow direction estimation, and anomaly detection (e.g., via trajectory clustering).
- Optimizing for edge deployment: Quantizing the model for faster inference on embedded devices like Jetson.
- Expanding dataset diversity: Incorporating additional datasets like VisDrone or Cityscapes for varied scenarios.
- Multi-camera fusion: Extending to track across multiple views for large-area surveillance.

These improvements would make the system more robust and versatile for real-world deployment.

References

[1] Gündüz, M., & Işık, E. (2023). A new YOLO-based method for real-time crowd detection from video. PMC.

[2] Enhanced Crowd Analysis Using YOLO. (2024). Zenodo.

[3] Ahmed, S., et al. (2023). Detection And Count of Human Bodies In a Crowd Scene Based on YOLO v5. ResearchGate.

[4] Evaluating Yolo Models for Detecting Crowds in Sparse Regions. (2025). Springer.

[5] Dense crowd detection method based on improved YOLOv7. (2024). SPIE.

[6] A pedestrian detection algorithm for low light and dense crowd. (2022). MATEC.

[7] Elharrouss, O., et al. (2024). Object detection and crowd analysis using deep learning. ScienceDirect.

[8] Crowd Detection Using YOLOv8. (2024). JETIR.

[9] Deep Learning-Based Crowd Surveillance and Density Estimation. (2024). IEEE.

[10] Aharon, N., et al. (2022). BoT-SORT: Robust Associations Multi-Pedestrian Tracking. arXiv.

[11] Improved BOTSORT multi-object tracking. (2025). ScienceDirect.

[12] Tracking Players in Volleyball Matches using Vol-Bot-SORT. (2025). IEEE.

[13] Dendorfer, P., et al. (2020). MOT20: A benchmark for multi object tracking in crowded scenes. arXiv.

[14] Multi-Target Tracking Based on a Combined Attention Mechanism. (2023). PMC.

[15] Bendali-Braham, M., et al. (2020). Revisiting crowd behaviour analysis through deep learning. PMC.

[16] Recent Deep Learning in Crowd Behaviour Analysis: A Brief Review. (2025). arXiv.

[17] Tripathi, G., et al. (2019). Intelligent video surveillance: a review through deep learning techniques for crowd analysis. ResearchGate.

[18] A review of deep learning-based crowd abnormal behavior detection. (2023). ACM.

[19] Recent trends in crowd management using deep learning. (2024). Springer.

[20] Deep convolutional neural network-based enhanced crowd density. (2025). Nature.