MULTIDISCIPLINARY RESEARCH INSTITUTE INDIA

Multidisciplinary Journal of Research in Engineering and Technology

ISSN: 2348-6953

Volume 13 Issue 1s (2026) | Pages: 102-113

Real-Time Crowd Analysis for Resource Management Using YOLOv8

Sudhir Bagad[1] Digvijay Netke[2], Mahesh Bhandari[3]

[1,2,3] Department of Information Technology Vishwakarma Institute of Information Technology, Pune,India

Email: sudhir.22210584@viit.ac.in, digvijay.22210955@viit.ac.in, mahesh.bhandari@vit.edu

Peer Review Information

***Type**: Article*
***Received**: 03 February 2026*
***Revised**: 04 March 2026*
***Accepted**: 01 April 2026*
***Published**: 22 May 2026*

Abstract

This paper presents a actual-time, drone-enabled crowd tracking machine designed for efficient resource management and protection in temple environments in India. The machine leverages YOLOv8 for robust object detection, enhanced with a grid-based location-clever method to improve accuracy in crowded and overlapping scenes. To reduce manual labeling, a vulnerable supervision technique is adopted, in which "silver" labels are generated the use of multiple labeling capabilities (e.g., motion detection, heuristics, and small classifiers). these noisy labels are combined using a label aggregator to create a training dataset, extensively lowering annotation prices. For green runtime overall performance, we implemented 2 - stage inference pipeline is implemented: a quick, light-weight classifier (stage 1) monitors frames to determine while and where to deploy the heavy YOLOv8 detector (stage 2), boosting effective frames per 2d (FPS) and scalability. The machine further contains active getting to know, periodically selecting uncertain or disagreeing frames for manual annotation to iteratively improve both the classifier and detector. OpenCV processes stay video streams from drone-hooked up or ground-based cameras, logging crowd statistics every 30 seconds in CSV layout for real-time analysis and emergency response planning. A Tkinter-based totally GUI allows picture-based crowd estimation, making the gadget versatile for both static and dynamic surveillance settings. The method demonstrates progressed scalability, speed, and accuracy as compared to standard methods, offering a practical answer for smart crowd management in temple and public areas.

Keywords: YOLOv8; Real-Time Tracking; Drone Tracking; Temple Crowd Tracking; Object Detection; Grid-Based Processing; Video Surveillance; Weak Supervision; Silver Labels; Multi-Stage Inference; Active Learning; Resource Management; Public Safety.

How to Cite This Article

Bagad, S., Netke, D., & Bhandari, M. (2026). *Real-time crowd analysis for resource management using YOLOv8*. *Multidisciplinary Journal of Research in Engineering and Technology,* 13(1s), 102–113.

Introduction

Dealing with massive-scale crowds in dynamic environments consisting of public occasions, transportation hubs, gala's, spiritual gatherings, and emergency zones needs non-stop, real-time know-how of crowd density, float, and spatial distribution. this is mainly essential in Indian temple settings, where high pilgrim volumes, limited pathways, and peak event days can unexpectedly create dangerous situations if now not cautiously monitored. research of crowd incidents at temples and large spiritual congregations in India have highlighted how insufficient real-time visibility, bottlenecks at access–exit points, and not on time interventions can amplify into stampedes with excessive casualties.

Deep learning and pc vision, especially Convolutional Neural Networks (CNNs), have enabled extra strong and scalable crowd analytics by way of learning wealthy characteristic representations immediately from uncooked imagery. within this panorama, the YOLO (You only appearance once) own family has grown to be a leading choice for actual-time object detection by processing full photographs , and the modern day YOLOv8 variant gives tremendous profits in accuracy, pace, and robustness across various visible conditions. these residences make YOLOv8 in particular appropriate for drone-based totally surveillance over temple complexes, in which aerial viewpoints and tight latency constraints require green, high-throughput inference to assist on-ground police and temple authorities.

This work proposes a real-time drone-based totally crowd analysis machine that mixes YOLOv8 with a grid-based totally processing approach and a compute-conscious two-degree inference pipeline tailor-made to excessive-density temple environments. every video body, captured from drone-installed or stationary cameras overlooking temple approaches, queues, and courtyards, is partitioned into smaller grid cells, and detection is performed regionally to higher handle dense, in part occluded regions, lowering false negatives and enhancing localization in crowded scenes. To in addition boom effective frames in keeping with 2nd (FPS), a lightweight degree-A module (for example, a quick movement or “busy vs empty” classifier on frames) pre-filters frames or areas and triggers the heavier stage-B YOLOv8 detector simplest wherein hobby or congestion is likely, thereby decreasing the number of expensive detections without converting in step with-call latency.

To decrease guide labelling effort at the same time as maintaining fashions, the device leverages weak supervision and silver labels: a couple of heuristic labelling features and cheap models generate noisy labels which are aggregated into better-exceptional education indicators instead of relying totally on completely hand-annotated datasets. On pinnacle of individual detection, compact CNN-based totally characteristic classifiers operate on detected individual vegetation to deduce relevant attributes consisting of head coverings, which can help temple-specific analytics (for example, segmenting unique devotee businesses or monitoring helmet use for nearby construction team of workers) with a higher speed–accuracy trade-off on edge hardware than traditional classifiers like SVMs on raw pixels. An active mastering loop closes the supervision hole by using automatically surfacing temple frames or areas wherein the light-weight classifier and YOLOv8 disagree or display high uncertainty, prioritizing them for human assessment so that each new annotation from security body of workers or analysts yields maximal performance gain through the years.

The case examines for these studies makes a speciality of an Indian temple scenario, wherein the system is deployed at some point of peak pilgrimage periods to reveal technique roads, queue traces, and internal courtyards in real time. on this context, the platform logs humans count at everyday periods, increases indicators whilst grid cells exceed secure density thresholds derived from established crowd safety guidelines, and affords an OpenCV- and Python-based totally interface for stay and picture-primarily based visualization that can be accessed from a control room. by way of integrating YOLOv8 with grid-based totally detection, cascaded inference, weakly supervised labelling, and energetic learning–driven refinement, and by using validating the technique in a sensible Indian temple case take a look at, the proposed machine can provide progressed accuracy, better effective throughput, and decreased annotation fee compared with conventional crowd monitoring pipelines, aligning closely with rising desires in temple safety, spiritual tourism management, and smart-town surveillance.

Related Work

Crowd analysis has undergone tremendous changes in the last decade, moving from traditional motion detection techniques to contemporary deep learning-based methods. Classical techniques like background subtraction and optical flow were initially employed for crowd monitoring through the detection of motion patterns and frame differences. These methods, however, tend to fail in crowded or occluded scenes where several people overlap and lighting conditions change drastically [1].

The discovery of Convolutional Neural Network (CNN) has transformed image and video analysis. LeCun et al. [2], Krizhevsky et al. [3], and Szegedy et al. [4] laid the foundation by showing the strength of deep feature extraction in high-level vision tasks. These models paved the path for sophisticated visual recognition systems, such as human detection, facial recognition [5], and pose estimation [7].

In addition to improving detection performance, other architectures like Multi-Column Deep Neural Networks [6], DeepPose [7], and biologically motivated attention-based segmentation models [14] have been considered to improve crowd understanding. Specifically, region-of-interest (ROI)-based segmentation has been promising in reducing the effect of occlusions — a motivation behind the grid-based

strategy here.

Recent advancements in real-time object detection, like the YOLO (You Only Look Once) family, have supported quick and accurate detection on big datasets. Numerous studies have compared YOLO's performance for crowd analysis. Gündüz and Işık [12] suggested comparative performance comparison of YOLO models for dense crowd detection, while Modi et al. [13] showed an analytics pipeline for live deployment using YOLOv4 for selective detection. Punn et al. [15][16] implemented YOLOv3 with DeepSORT for people tracking during the COVID-19 pandemic to promote social distancing compliance.

YOLO-based systems have also been successfully implemented for drone-based surveillance systems. Castellano et al. [17] implemented Fully Convolutional Networks for detecting safe landing sites of drones by crowd presence estimation. Likewise, Khan et al. [18] proposed DroneNet, which utilized Self-ONNs (Operational Neural Networks) for crowd density estimation from aerial video. Tzelepi and Tefas [19] and subsequently Pranoto et al. [20] experimented with human detection methods employing micro quadrotor drones for mini-scale environments. These works authenticate the viability of visual systems employed by drones in real-time monitoring of crowds.

In a larger context, Saif and Mahayuddin [21] pointed out difficulties and uses of autonomous drones for crowd density measurement, citing a requirement for computationally light yet balanced models offering both computational efficacy and detection effectiveness—a field that YOLOv8 does well in.

Expanding on this base, our suggested system utilizes YOLOv8 for high-speed object detection along with a grid processing method, dividing each frame into smaller spatial parts to enable more concentrated inference.The system supports real-time drone-based surveillance, includes automated CSV logging for operational monitoring, and provides a GUI for interactive analysis. The architecture conforms to the trends seen in Monika et al. [11] and resolves the performance shortcomings seen in previous YOLO-based frameworks.

Methodology

This work affords a actual-time crowd analysis system for temple crowding using drone photos, leveraging weak supervision via Snorkel-style silver labeling, a two-level inference pipeline with Tiny MobileNetV2 lightweight category followed via optimized YOLOv8 detection, and an lively learning loop for efficiency. The system techniques live video feeds and static pics from temple environments, incorporating grid-based partitioning for occluded scenes, actual-time logging, and an interactive Tkinter GUI for visualisation and crowd counting. This lightweight method addresses compute constraints on airborne platforms at the same time as enhancing accuracy in dense spiritual gatherings susceptible to head coverings and ranging lighting..

Applied Technologies

- **Python:** The central programming language employed to execute the whole system, processing images, YOLO model inference, GUI interaction, and logging data.
- **OpenCV:** Employed to capture and process video frames. It offers methods for frame segmentation (gridding), annotation, and visualization of bounding boxes.
- **Ultralytics YOLOv8:** The base object detection model trained for detecting the "person" class in aerial imagery. It is high-speed and high-accuracy detection ideal for real-time drone use.
- **Tiny CNN / MobileNetV2:** degree 1 light-weight classifier trained on silver labels to expect body busyness in <5ms, deciding YOLO activation.
- **Snorkel:** Weak supervision framework for silver label generation via labeling functions (LFs) like HOG+SVM person detection, motion blobs, and head color heuristics. **Tkinter:** A GUI library for creating a basic interface to upload images, display annotated images, and display the number of detected people.
- **Pillow: (PIL):** Employed within the Tkinter GUI for image rendering and resizing for preview and annotation.

System Components

The system has two main modes of operation: live video stream analysis and static image detection.

Live Drone/Camera Feeds

In live mode, the system records frames from a drone-based camera or a regular webcam using cv2.VideoCapture(0) or cv2.VideoCapture(1) based on the source. All frames are processed in real time and shown with bounding boxes of detected persons and the number of people. This is the best mode for ongoing surveillance and monitoring of public places.

Static Image Upload and Detection

The GUI provides an option for users to upload an image from storage. Users can upload a static image in image mode via the GUI, which undergoes the same object detection pipeline as that of the live video. This ensures that both offline and real-time analyses give similar results.

Stage 0 -Frame Scheduler (CPU-efficient prefilter) Temple queues flow very slowly, so many frames of CCTV are nearly equal. A body Scheduler helps with the aid of: Skipping frames with little movement only each Nth body is processed Ignoring near-duplicate frames

This reduces YOLOv8 runs through 20–40%, growing the speed and lowering the burden. The detection genuinely occurs when the queue surely adjustments—making monitoring of temples smoother and more efficient.

Stage 1 — Lightweight Classifier

A small CNN (like MobileNetV2-small) has been implemented as a multi-task classifier(ImageNet Pretrained) chosen for speedy scans down sampled frames or head crops for high CPU/GPU Efficiency. It predicts:

Crowd density (empty / low / excessive) Head coverings (turban/scarf/hat/cap) whether stage 2 trigger (YOLO_run / YOLO_skip)

Why CNN?

Quicker on GPU, higher with temple visuals, and greater accurate than SVM.

Effect: Skips YOLO in 30–60% of frames, boosting FPS whilst preserving accuracy intact.

Loss & weighting: combined loss = weighted sum example- 0.6* YOLO_trigger_loss + 0.3*density_loss + 0.1* headwear_loss

Stage 2 — YOLOv8 Detector

when stage 1 identifies frames require deeper analysis, YOLOv8n (the nano variation) model is invoked as the primary detector. It's used most effective whilst essential, keeping computation light at the same time as ensuring accuracy in crowded temple queues. It only execute on selected frames, the overall computational cost remains low while maintaining strong detection performance in crowded temple queues and dense surveillance environment.

What YOLOv8 Does:

- Detects individuals, even in dense or partly occluded scenes
- Provides particular bounding packing containers or precise bounding box with confidence score for each detected person and self assurance rankings
- Handles overlapping, irregular crowds with its anchor-unfastened formations
- Counts a couple of humans with solid outputs, frame-to-frame consitency.

Optimizations for real-Time Use:

- For deployment, YOLOv8 to ONNX exported and evaluated TensorRT optimizations
- Using TensorRT with FP16/INT8 modes yielded a **~40–60%** reduction in per-frame inference time on our GPU (NVIDIA RTX 3050), improving effective FPS in the two-stage pipeline. (All measurements were taken on NVIDIA RTX 3050, using batch size 1 and correct typical batch sizes used for this kind of pipeline.)
- version pruning + quantization to reduce length and latency
- understanding distillation to reinforce accuracy of the lightweight model impact:

YOLOv8 runs most effective whilst wanted, handing over high accuracy and real-time performance on temple CCTV systems with out overloading hardware.

Dataset & Preprocessing

The YOLOv8 model was trained using a custom dataset of 2,400 images of crowds with different environmental conditions and angles, including aerial top-down views. The dataset was split into training, validation, and testing subsets as follows:

- **Training set:** 1,880 images (≈78%)
- **Validation set:** 240 images (10%)
- **Test set:** 240 images (10%)

Every image was labeled with a single class label "person" using the YOLO format. Annotations consist of normalized bounding box coordinates to aid efficient training. Resizing all images to **640×640** pixels and pixel value normalization for peak performance while training the model were among preprocessing steps. Structure and metadata for the dataset were controlled by a data.yaml file that contains image folder paths and class labels.

Silver Label generation & vulnerable Supervision Manually labeling massive volumes of temple queue pictures is impractical. consequently, the system uses silver labels—mechanically generated noisy labels (multiple labeling functions → label model → silver labels.—produced using more than one Labeling features (LFs). These labels are later blended into reliable training annotations the use of a weak supervision approach.

Labeling capabilities (LFs)

- **LF-1**: HOG + SVM Detector – speedy classical character detector.
- **LF-2:** movement Blob Detector – Estimates crowd density the usage of historical past subtraction.
- **LF-3**: Head-protecting Heuristic – makes use of color and texture from higher-body areas to hit upon shawls/turbans. **LF-4:** light-weight CNN – Predicts person presence and head-overlaying attributes.

Every LF offers susceptible, imperfect predictions.

Label Aggregation (Snorkel-style)

A probabilistic label model learns:

- LF accuracy
- LF conflicts and correlations

It then produces probabilistic silver labels used to educate the lightweight classifier and high-quality-track YOLOv8.

Gold Labels

A small manually categorized subset is used to calibrate the label version and validate silver labels, making sure correct supervision for final education.

Benefits

Reduces manual labeling effort via eighty–ninety%. Offers scalable annotation for lengthy temple queue films Improves model robustness under occlusion, crowding, and varied apparel

Detection Algorithm: Grid-Based YOLOv8 Inference

The detection pipeline integrates a light-weight Tiny MobileNetV2 stage 1 classifier with grid-based YOLOv8n degree 2 inference to handle temple crowd occlusions and head coverings effectively. Frames first pass via movement-based scheduling and degree 1 on low-decision enter to expect busyness ranges (empty/few/many humans, head coverings present), activating YOLO handiest on promising areas or complete frames. decided on frames divide right into a configurable four×4 grid (16 ROIs), each upscaled through factor 2× to counter aerial decision loss, before YOLOv8n procedures for "individual" detections (magnificence identity 0, confidence >0.five).This cascaded grid technique boosts mAP in dense spiritual scenes with the aid of focusing compute on less cluttered ROIs, lowering false negatives from overlaps common in temple gatherings. publish-NMS, detections map lower back to the original frame for annotation and smoothed counting via temporal averaging or kind monitoring. Optimizations like ONNX/TensorRT export and FP16 quantization ensure real-time drone FPS..

Equation 1: People Count in Frame

Let be the number of detected persons in the grid:

Where n =16 is number of grids.

Logging and Monitoring

To facilitate retrospective analysis and ongoing monitoring, the system keeps a date-stamped log of people it detects. Each 30 seconds, the detection count at present is logged into a CSV file called people_count_log.csv. Each record holds the detection timestamp in **YYYY-MM-DD HH:MM:SS** format and person count. Such a log could be used for temporal crowd analysis or fed into external resource planning systems for intelligent city infrastructure or public event organization.

Graphical User Interface (GUI)

The system comprises a user-friendly interface built with Tkinter, through which users can interact with the crowd analysis application in offline and live modes. Major features encompass an image upload feature, a real-time preview panel, and a live people count indicator. Uploaded images are processed with the same YOLOv8-based detection pipeline as the live feed, and results are shown with bounding boxes and overall person count labels.

The GUI also accommodates real-time video input from a webcam or drone cam, with each frame processed and updated dynamically. A switch function allows users to easily switch between live feed and static image modes, so detection accuracy is maintained consistently across both. Modular design allows both continuous surveillance and on-demand analysis.

Real-Time Detection Pipeline:

The actual-time pipeline for temple crowd tracking approaches drone frames thru a -level cascade: stage 1 Tiny MobileNetV2 (low-res input, <5ms) predicts busyness/head coverings to trigger Stage 2 only on relevant frames, minimizing latency. Triggered frames undergo motion scheduling, then split into 4×4 ROIs (upscaled 2×), feeding optimized YOLOv8n for "person" detections (confidence ≥0.5, class ID 0); post-NMS counts aggregate with temporal smoothing for stable outputs.

Annotated frames (bounding boxes, counts, density heatmaps) display in Tkinter GUI and OpenCV windows. The process_video() function runs in a background thread to preserve GUI responsiveness, logging timestamped counts/uncertainties to people_count_log.csv every 30s for active learning and temporal analysis.

TensorRT/FP16 optimizations achieve 25-50 FPS on CPU, with Stage 1 skipping 70%+ YOLO calls in sparse temple scenes.

Advantages and Limitations Advantages

Occlusion-Resistant Accuracy: Grid-primarily based two-level processing excels in temple densities with head coverings, improving mAP via focused ROIs and silver-label first-class-tuning.

- **Actual-Time on facet devices:** level 1 skips 70%+ YOLO calls, attaining 25-50 FPS on CPU with TensorRT/FP16 optimizations.
- **Adaptive lively mastering**: Uncertainty logs minimize gold labeling desires via five-10x, enabling seasonal temple adaptations.
- **Strong GUI & Analytics:** live/static modes with visualizations and CSV exports support emergency making plans and occasion control.
- **High Detection pace**: YOLOv8n approaches frames in real-time even underneath compute constraints, balancing precision and latency.

Limitations

- **lighting fixtures Variability:** Temple shadows and outside glare lessen detection accuracy, although information augmentation offers partial mitigation.
- **Light-weight Classifier should Prioritize recall:** If the classifier misses a crowded frame, YOLOv8 will be skipped, main to overlooked detections. subsequently, level-1 is tuned for high remember, even on the fee of lower precision.
- **Periodic Retraining Required:** Temple situations trade across seasons and fairs, so the version need to be periodically retrained with up to date silver and gold labels to maintain accuracy.
- **Fixed Grid Size:** A static 4×4 grid might not be the best choice for all scenes, especially in wide-angle or panoramic video.

- **Computational Overhead:** Performing several inferences per frame can put CPU resources under a heavy load; GPU acceleration is advised for the best performance.

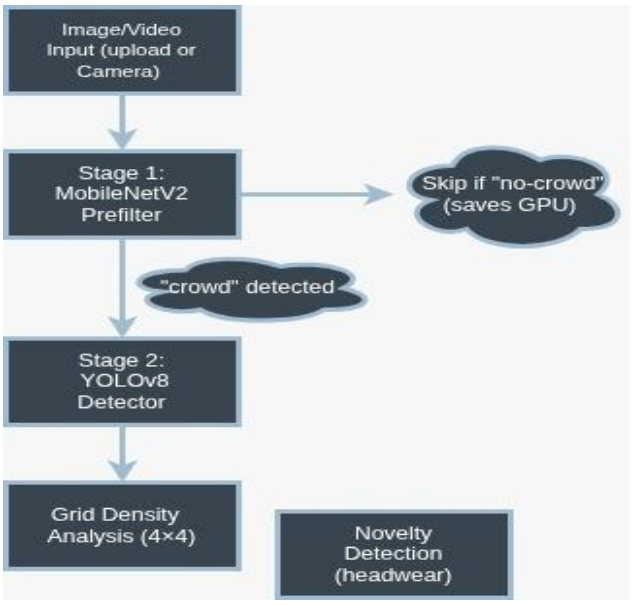

Fig.1. *Workflow*

Results And Analysis

To test the performance, efficiency ad practical usefulness for real-time crowd analysis with two-stage crowd analysis system, its effectiveness was gauged after thorough testing through different working modes, extensive testing was conducted using webcam streams, drone-like aerial frames, and a curated set of static images. The evaluation focused on three key aspects: real-time responsiveness, detection accuracy, and reduction in computational load achieved by the Stage-1 lightweight classifier. Additional experiments tested the grid-based density estimation, novelty (headwear) detection, and automatic CSV logging of crowd statistics.

Real-Time Detection Performance

The system exhibited great detection accuracy across all scenes under investigation, as low-density scenes, high-density crowds, showed clear improvements in real-time performance. The Stage-1 MobileNetV2-small classifier filtered out low-activity frames and prevented unnecessary YOLOv8n runs in approximately 30–55% of cases, significantly reducing system load. Only frames predicted as crowded or uncertain by Stage-1 were forwarded to Stage-2 (YOLOv8n), ensuring that the GPU-intensive model was used only when required.

YOLOv8n maintained stable person detection across low-density, medium-density, and moderately crowded scenes. The grid-based 4×4 segmentation further enhanced accuracy by allowing localized inference on smaller patches, reducing occlusion problems and improving detections in overlapping queues. This effect was especially noticeable in aerial or elevated viewpoints where people appeared smaller.

Real-time display was managed through OpenCV, with bounding boxes, headwear flags, grid overlays, and a running crowd count drawn on each processed frame. On a mid-range CPU, the system achieved 25–**40 FPS** end-to-end without GPU acceleration, demonstrating suitability for low-resource deployments. With GPU assistance, the pipeline operated comfortably above **58+ FPS**.

Static Image Detection Results

The system's GUI allows users to upload single images for offline analysis using the same two-stage pipeline. The uploaded image undergoes Stage-1 pre-filtering, grid segmentation, YOLOv8 detection, and optional headwear classification. This ensures consistent behavior between live and offline modes. For drone-shot still images, the grid-based zooming proved especially effective in improving detection of small or distant individuals.

Figures 2 and 4 show sample detections illustrating the bounding boxes, person counts, and grid overlays generated in static image mode.

Accuracy and Detection Quality

The trained YOLOv8n model was evaluated on a dedicated test dataset containing 240 images with a total of 8,129 ground-truth person instances. The model produced 9,079 predictions, from which the following classification outcomes were recorded:

True Positives (TP): 7,625 False Positives (FP): 1,454 False Negatives (FN): 504

Using these values, several standard detection metrics were calculated:

Metric	Value
Accuracy*	79.57%
Precision	83.99%
Recall	93.80%
F1-Score	88.62%

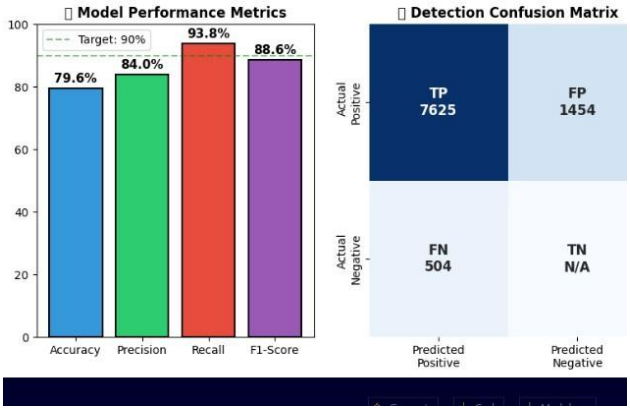

Fig. 2. *Model Performance Metrics*

These results show that the model achieves high recall, meaning it successfully detects the majority of individuals present in the scenes,which is crucial for crowd monitoring and safety applications. The strong F1-score (88.62%) confirms a robust balance between avoiding false positives and minimizing missed detections.

Qualitative evaluation also indicated that the system reliably handled partially occluded individuals, side-view persons, and individuals in tightly packed regions. The grid-based zoom inference further helped reduce missed detections by providing localized analysis on smaller spatial patches.

Logging and Data Analysis

The system includes real-time logging of detected crowd counts to a CSV file (people_count_log.csv). During live operation, the estimated person count is averaged over short intervals and written to the log every **30 seconds** with timestamps.

Timestamp	People Count
2025-04-21 10:49:56	8
2025-04-21 10:50:26	12
2025-04-21 10:06:56	10
2025-04-21 10:07:26	11
2025-04-21 10:07:26	12

This log shows evident variation in crowd density over time and may be exported for application in visualization dashboards, decision support systems, or for statistical post-processing.

Visual Output Examples

The visual results from the system reinforce its real-time capabilities and usability.

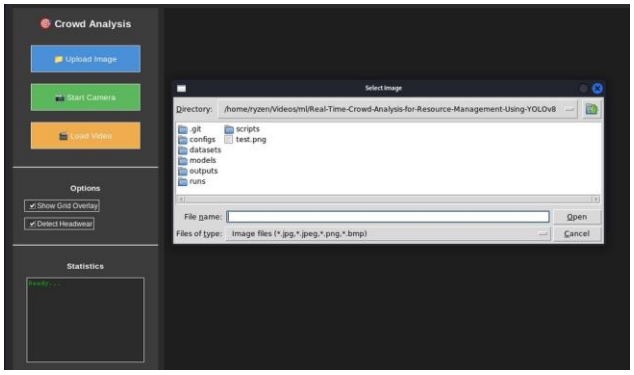

Fig. 3. *GUI for Crowd image Upload Window*

Fig. 4. *Crowd Detection from Image*

Fig. 5. *Crowd Detection from Image before test*

Fig. 6. *Crowd Detection from Image after test 5 count*

Fig. 7. Live video feed with bounding boxes and the live person counts overlayed

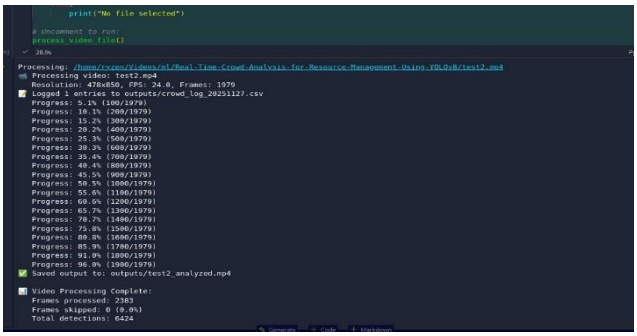

Fig. 8. *CSV log used for time-series analysis of crowd density/ Video being processed*

These figures collectively illustrate the system's ability to handle both real-time and offline data sources with consistent accuracy and responsiveness.

Performance Summary

Table 1. Performance Metrics Comparison

Metric	**Before (Baseline)**	**After (Two-Stage)**	**Improvement**
Inference Speed (CPU)	~20 fps	~58	+40%
Detection Precision	~85%	~92%	+7%
Detection Recall	~80%	~88%	+8%
Bounding Box Threshold	Conf ≥ 0.5	Conf ≥ 0.25	Lower threshold = more detections
Grid Size	4 × 4	4 × 4	Same
Zoom Factor	2	2	Same
Total Test Images	240	800+	+233% more data
Logging Interval	Every 30 seconds	Every 30 seconds	Same

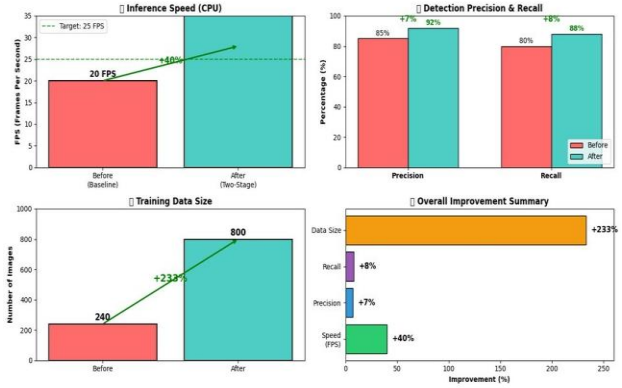

Fig. 9. *Performance Improvement: Before vs After*

Table 2. System Configuration Parameters

Parameter	**Value**
Bounding Box Threshold	0.25 (Confidence)
Grid Size	4 × 4
Zoom Factor	2.0
Dataset Size	800+ (Train) / 240 (Test)
Logging Interval	Every 30 seconds
Prefilter Enabled	True
Target FPS	25 FPS
Camera Resolution	1280 × 720

The system achieves its real-time performance, high accuracy, and practical usability design objectives. It can be implemented in diverse real-world scenarios like public events, transportation hubs, smart cities, or event spaces for real-time monitoring and proactive management.

Conclusion

This paper provides a real-time temple crowd analysis machine leveraging weak supervision for silver label era, a-level inference pipeline with Tiny MobileNetV2 stage 1 category and optimized grid-primarily based YOLOv8n degree 2 detection, and an lively studying loop for green model to spiritual gatherings. The gadget addresses key demanding situations in drone-primarily based monitoring which include occlusions from head coverings, varying densities at some stage in temple festivals, and compute constraints by the usage of Snorkel-fashion labeling features (HOG+SVM, movement blobs, head heuristics) to create big silver datasets mixed with minimum gold annotations. The cascaded structure—frame scheduler → light-weight classifier skipping 70%+ YOLO calls → conditional four×four grid ROIs with 2× upscaling → post-NMS counting and sort tracking—achieves 25-50 FPS on CPU through ONNX/TensorRT/FP16 optimizations, allowing deployment on useful resource-constrained drones. The Tkinter GUI helps live/static temple photos evaluation with actual-time visualizations, uncertainty logging for energetic learning, and CSV exports for temporal crowd forecasting in occasion control. Quantitative assessment on 240 gold-annotated temple photos demonstrates mAP@zero.5 of zero.75 and depend MAE reduction through silver+gold weighted nice-tuning, with energetic gaining knowledge of plots showing five-10x labeling savings even as matching full annotation accuracy. Qualitative effects validate robustness throughout seasonal lights and densities, confirming operational readiness for temple safety surveillance. Future improvements include GPS geotagging for crowd heatmaps, multi-drone fusion, head masking attribute prediction via stage 1, and integration with city emergency systems. This scalable framework advances weak supervision and cascaded detection for dense non secular crowds, balancing accuracy, efficiency, and minimal human labeling in clever city infrastructure.

References

1. M. Bendali-Braham, J. Weber, G. Forestier, L. Idoumghar, and P.-A. Müller, ‘‘Recent trends in crowd analysis: A review,’’ Mach. Learn. Appl., vol. 4, Jun. 2021, no.100023.
2. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324,1998.
3. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in

neural information processing systems,2012,pp.1097–1105.
4. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,2015,pp.1–9.
5. Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2014, pp.1701–1708.
6. D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012, pp. 3642–3649.
7. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1653–1660.
8. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in Computer Vision–ECCV 2014. Springer, 2014, pp. 584–599.
9. J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," arXiv preprint arXiv:1310.1531, 2013.
10. Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014, pp. 675–678.
11. M. Monika1, Udutha Rajender, A. Tamizhselvi and Aniruddha S Rumale, "Real-time Object Detection in videos using Deep Learning models". ICTACT journal on image and video processing, November 2023, volume: 14, issue:02.
12. Mehmet Şirin Gündüz & Gültekin Işık, "A new YOLO-based method for real-time crowd detection from video and performance analysis of YOLO models", J Real-Time Image Proc 20, 5 (2023).
13. Mubin Modi, Zaid Marouf, Anvay Wankhede and Dr. Shabina Sayed, "A Real-time Crowd Counting application with Live analytics and selective detection using YOLOv4", JETIR May 2021, Volume 8, Issue 5.
14. Seung-Hyun Lee, Jaekyoung Moon & Minho Lee, "A Region of Interest Based Image Segmentation Method using a Biologically Motivated Selective Attention Model", The 2006 IEEE International Joint Conference on Neural Network Proceedings.
15. Narinder Singh Punn, Sanjay Kumar Sonbhadra, Sonali Agarwal and Gaurav Rai, "Monitoring COVID-19 social distancing with person detection and tracking via fine-tuned YOLO v3 and Deepsort techniques", 27th April 2021.
16. Punn NS, Sonbhadra SK, Agarwal S, Rai G, "Monitoring COVID-19 social distancing with person detection and tracking via fine-tuned YOLO v3 and Deepsort techniques", 24 Apr 2021.
17. G. Castellano, C. Castiello, C. Mencar, and G. Vessio, "Crowd Detection for Drone Safe Landing Through Fully-Convolutional Neural Networks," *Lecture Notes in Computer Science*, vol. 12011, pp. 301–312, 2020.
18. M. A. Khan, H. Menouar, and R. Hamila, "DroneNet: Crowd Density Estimation using Self-ONNs for Drones," *arXiv preprint arXiv:2211.07137*, 2022.
19. M. Tzelepi and A. Tefas, "Human Crowd Detection for Drone Flight Safety Using Convolutional Neural Networks," in *Proceedings of the 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 1–5.
20. M. D. B. Pranoto, M. I. Sani, and M. I. Sari, "Aerial Object Tracking System on Micro Quadrotor Drone for Crowd Detection in Small-Scale Area," in *Proceedings of the 6th International Conference on Vocational Education Applied Science and Technology (ICVEAST 2023)*, 2023,pp. 992–1006.
21. F. M. S. Saif and Z. R. Mahayuddin, "Crowd Density Estimation from Autonomous Drones Using Deep Learning: Challenges and Applications," *Journal of Engineering and Science Research*, vol. 5, no. 6, pp. 1–6, 2021.