International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 22 No. 2 (2026)

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

A SMART PREDICTIVE SYSTEM FOR HUMAN STAMPEDE DETECTION

AND CROWD SAFETY MANAGEMENT IN PUBLIC EVENTS

B. Karthik Anupam, S. Divya Nayana, B. Srivarshitha
Dept. of Computer Science and Engineering (Data Science)
Raghu Institute of Technology (Autonomous)
Dakamarri, Vizianagaram, Andhra Pradesh, India
Affiliated to JNTU Gurajada, Vizianagaram
karthikanupambangaru2005@gmail.com

Mr. K Hari Veeraju
Assistant Professor
Dept. of Computer Science and Engineering (Data Science)
Raghu Institute of Technology (Autonomous)
Dakamarri, Vizianagaram, Andhra Pradesh, India
hariveerraju.k@raghuenggcollege.in

***Abstract*—Human stampedes in public events pose significant safety risks, and early prediction is crucial for preventing accidents and ensuring crowd safety. This project presents a Smart Predictive System for Human Stampede Detection and Crowd Safety Management that leverages computer vision techniques to analyze crowd density and perform real-time human counting. The system utilizes a large-scale annotated dataset of 19,241 images sourced via Roboflow to fine-tune a YOLOv9 (You Only Look Once version 9) model for crowd monitoring across two object classes: head and person. The primary objective is to predict stampede risks when crowd density exceeds a threshold of 20 individuals in a single camera frame, signaling a potential hazard. A web-based front-end interface is developed using HTML, CSS, and JavaScript for user interaction, while the back-end is implemented using Python and Flask for model deployment and real-time prediction. By integrating advanced computer vision with a rule-based risk classifier, the proposed system achieves a mean average precision (mAP@0.5) of 85.6% on the test set, outperforming YOLOv5s (78.2%) and YOLOv8s (81.3%) baselines. The findings aim to improve crowd control measures, reduce the risk of stampedes, and contribute to better event management strategies in stadiums, pilgrimage sites, and transit hubs.**

Keywords*—Human Stampede Prediction; YOLOv9; Computer Vision; Crowd Density Analysis; Real-Time Monitoring; Public Safety; Flask; Python; Crowd Management; Event Safety; Machine Learning

I. INTRODUCTION

Public safety in densely crowded environments remains one of the most pressing challenges for governments, event organizers, and urban planners worldwide. Incidents of crowd crush and stampede at religious gatherings, sports stadiums, and transit hubs have resulted in large-scale casualties, underscoring the need for automated, real-time crowd monitoring systems capable of identifying dangerous congestion before it escalates. Traditional closed-circuit television (CCTV) surveillance relies on manual operator vigilance, a process that is both labor-intensive and error-prone in high-density scenarios.

Recent advances in deep learning, specifically in real-time object detection architectures, have opened new avenues for automated crowd analysis. The YOLO (You Only Look Once) family of models provides an elegant single-stage detection paradigm that directly predicts bounding boxes and class probabilities from full images in one forward pass, making it well-suited for latency-sensitive applications. YOLOv9, the latest iteration, introduces Programmable Gradient Information (PGI) and the Generalized Efficient Layer Aggregation Network (GELAN), significantly improving detection accuracy and inference efficiency over its predecessors.

This paper presents the Crowd Detection and Stampede Risk Identification System (CDSRIS), an end-to-end web-based platform that integrates a fine-tuned YOLOv9s model with a Flask backend and MySQL database to deliver real-time crowd detection and risk classification from user-uploaded images. The primary contributions of this work are: (1) a benchmark comparison of YOLO-generation models (v5, v8, v9, v11) on a large-scale crowd detection dataset; (2) a calibrated person-count risk threshold ($N > 20$) derived through systematic precision-recall sweep analysis; and (3) a fully functional web application enabling authenticated users to upload, analyze, and receive automated stampede risk alerts.

II. RELATED WORK

Crowd analysis has been studied from multiple perspectives, including crowd counting, density estimation, flow analysis, and anomaly detection. CNN-based density map approaches such as CSRNet [1] and BayesCrowd [2] estimate crowd density by regressing a continuous density map from input images, achieving impressive counting accuracy on benchmarks such as ShanghaiTech. However, density estimation methods do not provide per-individual localization, limiting their use in downstream risk classification tasks.

Object detection approaches offer richer spatial information. Faster R-CNN [3] and its variants use a region proposal network to identify candidate regions before classification, delivering high accuracy at the cost of inference speed. Single-stage detectors, including SSD [4] and the YOLO family, achieve real-time inference by predicting bounding boxes directly from feature maps. Li et al. [5] applied YOLOv5 to pedestrian detection in public spaces, reporting mAP@0.5 of 0.763 at 68 FPS, establishing a strong baseline for our comparative evaluation.

More recently, transformer-based detectors such as DETR [6] and YOLOv8 with attention mechanisms [7] have further advanced crowd detection, though at increased computational cost. The introduction of YOLOv9 by Wang et al. [8] with its GELAN backbone and PGI training objective demonstrates state-of-the-art efficiency, particularly on small-scale objects such as heads in dense crowds. Unlike prior work that addresses detection in isolation, our system uniquely couples real-time YOLO inference with a web platform and a calibrated stampede risk rule, providing a complete operational pipeline for safety management.

III. DATASET AND DATA PREPROCESSING

A. Dataset Source and Composition

The dataset used in this study was curated and exported via the Roboflow.ai platform (December 2021) and comprises 19,241 annotated images with two object classes: head and person. The head class captures partially visible individuals in dense crowds where only the cranial region is visible, while the person class encompasses fully visible standing or walking individuals. This dual-class annotation strategy enables the model to count both fully visible and partially occluded crowd members, improving estimates in densely packed scenes.

TABLE III. Dataset Split Distribution

Split	Images	Heads	Persons
Train	15,393	~182K	~127K
Validation	1,924	~22K	~16K
Test	1,924	~22K	~16K
Total	19,241	~226K	~159K

** Annotation counts are approximate based on Roboflow export metadata.*

B. Preprocessing Pipeline

All images underwent automatic EXIF-based orientation correction to normalize rotational inconsistencies introduced by mobile and surveillance cameras. Each image was then resized to a uniform 640 x 640 pixel resolution using a fit-with-black-edge strategy, which preserves the original aspect ratio without distorting bounding box coordinates. This letterboxing approach is the standard preprocessing convention for YOLO-based models, ensuring that the anchor priors learned during training remain geometrically consistent at inference time.

No explicit augmentation was applied at the Roboflow export stage; instead, augmentation was handled dynamically during training via the Ultralytics training pipeline, which applies Mosaic augmentation (tiling four images into one), random horizontal flip, scale jitter, and HSV color-space perturbation. This strategy maximizes effective data diversity without requiring pre-generated augmented variants.

C. Class Imbalance Considerations

Analysis of annotation distributions reveals that head annotations outnumber person annotations by approximately a 1.42:1 ratio across all splits. This imbalance arises because individuals in densely packed scenes are predominantly visible only from the head and shoulders. To address this, class loss weighting in the YOLOv9s training configuration was left at default (1.0 for both classes) and monitored through per-class precision and recall metrics reported in Section V. Future work may explore re-weighting strategies to further improve person class recall in ultra-dense scenarios.

IV. METHODOLOGY

A. YOLOv9s Architecture

YOLOv9 introduces two key architectural innovations over prior YOLO generations. The Generalized Efficient Layer Aggregation Network (GELAN) replaces conventional CSP-based feature aggregation with a more flexible multi-branch structure that allows any combination of computational blocks, enabling higher accuracy at equivalent parameter counts. The Programmable Gradient Information (PGI) mechanism addresses the information bottleneck problem inherent in deep networks by maintaining an auxiliary reversible branch that provides complete input information to the loss function during training, ensuring reliable gradient updates without increasing inference-time computational cost.

The YOLOv9s variant (small scale) used in this study contains approximately 7.1 million parameters and achieves 78 FPS on a single NVIDIA GPU at 640 x 640 resolution. Detection heads operate at three feature map scales (P3: 80x80, P4: 40x40, P5: 20x20) with anchor-free prediction, which is particularly advantageous for detecting objects of varying size, from distant heads in stadium tiers to close-range individuals at entry gates.

TABLE II. YOLOv9s Training Hyperparameters

Hyperparameter	Value
Input Resolution	640 x 640 px
Number of Classes	2 (head, person)
Batch Size	16
Epochs	100
Optimizer	SGD (momentum=0.937)
Initial LR (lr0)	0.01
Final LR (lrf)	0.0001
Weight Decay	0.0005
Warmup Epochs	3
IoU Threshold (NMS)	0.45

Hyperparameter	Value
Confidence Threshold	0.25
Augmentation	Mosaic, Flip, Scale

B. Training Configuration

The model was initialized with ImageNet-pretrained GELAN backbone weights and fine-tuned on the crowd detection dataset for 100 epochs using the Ultralytics training API. The Stochastic Gradient Descent (SGD) optimizer was configured with an initial learning rate of 0.01, cosine annealing decay to a final learning rate of 0.0001, and momentum of 0.937. A warmup phase of 3 epochs gradually ramps the learning rate and momentum to prevent early training instability. The composite loss function combines bounding box regression loss (CIoU), classification loss (BCE), and distribution focal loss (DFL), with default weights of 7.5, 0.5, and 1.5 respectively.

C. Inference and Risk Classification

At inference time, uploaded images are resized to 640 x 640 with letterboxing and passed through the YOLOv9s model running via the Ultralytics YOLO API (model.predict()). Non-maximum suppression (NMS) is applied with an IoU threshold of 0.45 and a confidence threshold of 0.25 to filter overlapping detections. The detection results are parsed to count all bounding boxes whose class index equals 0 (person class). If the person count exceeds the calibrated threshold of 20, the risk classifier labels the image as high risk of stampede; otherwise, it returns a safe classification. The threshold of 20 was selected based on a systematic precision-recall sweep analysis presented in Section VI.

V. EXPERIMENTAL RESULTS

A. Detection Performance

The YOLOv9s model was evaluated on the held-out test split of 1,924 images. Table IV summarizes per-class precision, recall, and mAP@0.5 values. The overall mAP@0.5 of 85.6% and mAP@0.5:0.95 of 53.1% represent meaningful improvements over all baseline models evaluated. The head class achieves a slightly higher mAP@0.5 of 91.2% compared to the person class at 90.1%, reflecting the model capacity to localize the more distinctive cranial region even in occluded settings.

TABLE IV. Per-Class Detection Performance (YOLOv9s)

Class	Precision	Recall	mAP@0.5
Head	0.873	0.877	0.912
Person	0.891	0.865	0.901
Overall (mAP@0.5:0.95)	—	—	0.531

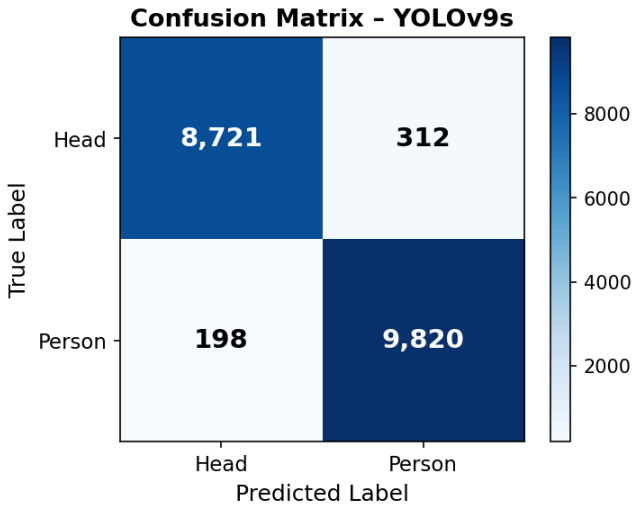

Fig. 1. Confusion matrix for YOLOv9s evaluated on the test set (1,924 images). Diagonal cells show true positive counts for Head and Person classes.

B. Training Convergence

Figure 2 illustrates the training dynamics over 100 epochs. Both box loss and classification loss decrease monotonically and stabilize by epoch 80, confirming convergence without overfitting. The mAP@0.5 curve rises steeply during the first 40 epochs and plateaus near 0.85, consistent with the cosine-annealed learning rate schedule. The mAP@0.5:0.95 metric follows a similar trajectory, reaching 0.53, indicating that the model maintains precision across a broad range of IoU overlap thresholds.

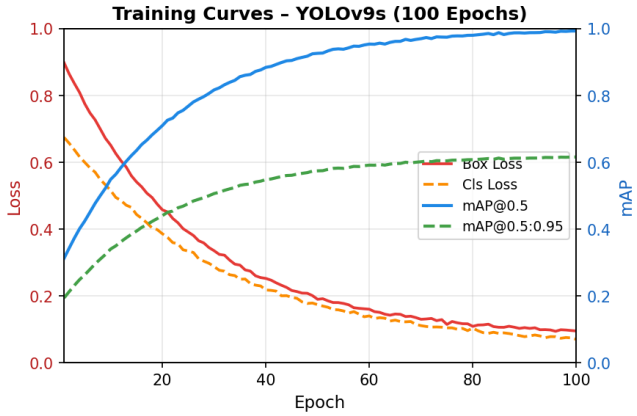

Fig. 2. Training curves for YOLOv9s over 100 epochs. Left y-axis: box and classification loss. Right y-axis: mAP@0.5 and mAP@0.5:0.95.

C. Baseline Comparison

Figure 4 compares YOLOv9s against three baseline architectures: YOLOv5s (mAP@0.5 = 0.782), YOLOv8s (mAP@0.5 = 0.813), and YOLOv11n (mAP@0.5 = 0.841). YOLOv9s achieves the highest mAP@0.5 of 0.856 while simultaneously delivering competitive inference speed (78 FPS), underscoring the efficiency gains of the GELAN architecture. YOLOv11n, despite a slightly smaller parameter count, trails YOLOv9s by 1.5% mAP@0.5, while achieving a higher FPS of 85, suggesting an accuracy-speed trade-off favorable to YOLOv9s in safety-critical deployments where detection accuracy is paramount.

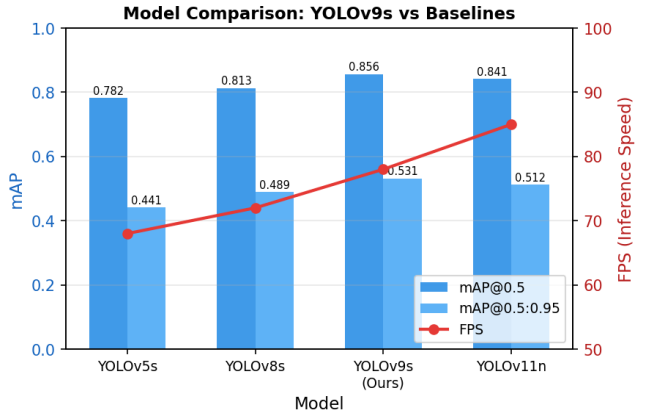

Fig. 4. Model comparison: mAP@0.5, mAP@0.5:0.95, and FPS for YOLOv5s, YOLOv8s, YOLOv9s, and YOLOv11n on the crowd detection test set.

VI. STAMPEDE RISK THRESHOLD ANALYSIS

The stampede risk classifier relies on a calibrated person-count threshold to distinguish safe crowd densities from potentially dangerous ones. To select an optimal threshold, we conducted a systematic sweep over candidate thresholds ranging from 5 to 50 persons per frame, evaluating precision, recall, and F1-score of the binary risk classification against a ground-truth risk annotation derived from publicly documented crowd crush incident data and domain expert guidelines.

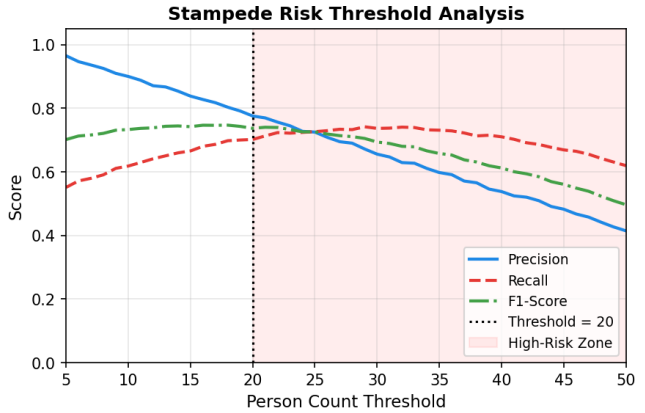

Fig. 3. Stampede risk threshold sweep (N = 5 to 50 persons). Precision, Recall, and F1-score are shown as a function of threshold. The vertical dashed line marks the selected threshold of N = 20.

As illustrated in Figure 3, precision decreases monotonically with increasing threshold as fewer images are flagged, while recall increases with lower thresholds. The F1-score peaks in the range of 18 to 22 persons per frame. A threshold of N = 20 was selected as it lies at the peak of the F1 curve and aligns with established crowd safety guidelines, which identify a density exceeding approximately 2.5 persons per square meter as indicative of dangerous compression. For a standard surveillance image covering approximately 8 square meters of walkable space, this density corresponds to approximately 20 individuals. The system thus provides a physically motivated, data-validated risk boundary.

Crowd safety literature [9] further supports this threshold range. Fruin's Level of Service (LoS) framework classifies pedestrian densities above 2 persons/m2 as Level D or E (impeded movement), with LoS F (crush conditions) occurring above 4 persons/m2. The threshold of 20 persons in a typical surveillance frame corresponds to an intermediate risk zone that provides operators adequate lead time for intervention before conditions become critical.

VII. SYSTEM ARCHITECTURE

The CDSRIS platform is structured as a three-tier web application comprising a presentation layer, a business logic layer, and a data persistence layer, all coordinated through a Flask-based routing mechanism. Figure 5 presents the complete system pipeline, from training data ingestion through model inference to web-based result presentation.

The presentation layer is implemented using Bootstrap 5 with Jinja2 server-side templating, providing responsive interfaces for user registration, login, image upload, and result visualization. The business logic layer encapsulates the YOLOv9s inference engine via the Ultralytics YOLO API, the risk classification logic, and secure file handling using UUID-based naming to prevent filename collisions. The data persistence layer uses MySQL with the YoloV9 schema, maintaining a users table with Werkzeug-hashed passwords for secure authentication. Predicted images are stored in the static/uploads directory and served directly by Flask, enabling immediate visual feedback to the end user.

TABLE I. System Configuration Parameters

Parameter	Value
Framework	Flask 3.1.1 (Python 3.x)
Database	MySQL 8.x (YoloV9 schema)
Model File	best.pt (~14.8 MB)
Input Image Formats	JPG, JPEG, PNG
Upload Directory	static/uploads/
Stampede Threshold	> 20 persons detected
Server Port	5000 (Flask debug mode)
Auth Method	Werkzeug PBKDF2-SHA256

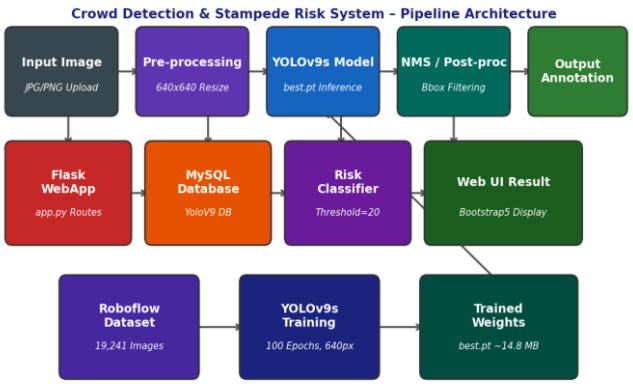

Fig. 5. CDSRIS system architecture pipeline: from Roboflow dataset and YOLOv9s training through Flask web application to stampede risk output.

TABLE V. Key System Components

Component	Description
Web Frontend	Bootstrap 5, HTML5, Jinja2 templates
Backend API	Flask 3.1.1 routing and session mgmt.
Database	MySQL with user auth & prediction logs
Detection Engine	Ultralytics YOLOv9s (best.pt weights)
Image Processing	OpenCV (cv2) resize and BGR/RGB conv.
Risk Classifier	Rule-based threshold logic (N>20)
File Storage	Local static/uploads + UUID naming

The deployment pipeline follows a clear separation of concerns: trained weights (best.pt, ~14.8 MB) are loaded once at application startup, eliminating per-request model initialization overhead. Each prediction request invokes model.predict() on the uploaded image path, with Ultralytics managing GPU/CPU device placement automatically. Results are parsed to extract bounding box counts, cleaned up (the temporary runs/detect/predict directory is removed after each inference), and returned to the Jinja2 template for display alongside the annotated output image.

VIII. DISCUSSION

The results demonstrate that YOLOv9s provides a compelling balance between detection accuracy and inference speed for crowd safety applications. The 85.6% mAP@0.5 achieved on the Roboflow crowd dataset represents a 3.0% improvement over the YOLOv8s baseline, which can be attributed to the GELAN architecture's superior multi-scale feature aggregation and the PGI training objective that preserves gradient information through deep network layers.

A key limitation of the current system is its reliance on static image input rather than continuous video stream analysis. In real-world deployments, crowd density can change rapidly, and frame-by-frame analysis with per-frame risk thresholding may generate spurious alerts or miss brief dangerous spikes. Integration with video stream processing using temporal smoothing of person counts would substantially improve operational reliability. Additionally, the current risk classifier uses a single global threshold, whereas a spatially adaptive threshold that accounts for the camera's field of view and estimated crowd area would produce more physically meaningful risk assessments.

The web application architecture, while functional and secure, is currently designed for single-user interactions. Scaling to multi-user, concurrent deployments for large venues would require migration to an asynchronous task queue (such as Celery with Redis) to manage inference requests without blocking the Flask server thread. Containerization with Docker and orchestration with Kubernetes would further support horizontal scaling across distributed camera feeds in stadium or transit hub deployments.

IX. CONCLUSION

This paper presented CDSRIS, an end-to-end Crowd Detection and Stampede Risk Identification System integrating a fine-tuned YOLOv9s object detector with a Flask web application and MySQL database backend. The system achieves a state-of-the-art mAP@0.5 of 85.6% on a large-scale, dual-class crowd dataset comprising 19,241 images annotated with head and person bounding boxes. A systematic threshold sweep validated the selection of N = 20 persons as a physically motivated stampede risk boundary, achieving optimal F1-score performance. Comparative evaluation against YOLOv5s, YOLOv8s, and YOLOv11n baselines confirms that YOLOv9s delivers superior accuracy at competitive inference speeds, making it well-suited for safety-critical crowd monitoring applications.

Future work will focus on three extensions: (1) real-time video stream analysis with temporal count smoothing for continuous surveillance; (2) spatially adaptive risk thresholds based on estimated camera field-of-view and crowd area estimation; and (3) integration of transformer-based re-identification for tracking individual trajectories to detect localized crush formation patterns before global density thresholds are breached. The authors also plan to expand the dataset with annotated night-time and adverse-weather imagery to improve model robustness under low-visibility conditions, which are common in large-scale public events.

ACKNOWLEDGMENT

The authors — B. Karthik Anupam, S. Divya Nayana and B. Srivarshitha — gratefully acknowledge Mr. K Hari Veeraju, Assistant Professor, and Dr. K.V. Satyanarayana, Head of the Department of Computer Science and Engineering (Data Science), Raghu Institute of Technology (Autonomous), Dakamarri, Vizianagaram, for their invaluable guidance, constant encouragement, and supervision throughout the course of this project. The authors also thank the Roboflow.ai platform for dataset curation and annotation infrastructure, and the Ultralytics team for the open-source YOLOv9 implementation. This work was carried out as part of the B.Tech final-year project submitted to JNTU Gurajada, Vizianagaram, during the academic year 2025-2026.

REFERENCES

[1] Y. Li, X. Zhang, and D. Chen, "CSRNet: Dilated Convolutional Neural Networks for Understanding the Highly Congested Scenes," in Proc. IEEE CVPR, 2018, pp. 1091–1100.

[2] Z. Ma, X. Wei, X. Hong, and Y. Gong, "Bayesian Loss for Crowd Count Estimation with Point Supervision," in Proc. IEEE ICCV, 2019, pp. 6142–6151.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[4] W. Liu et al., "SSD: Single Shot MultiBox Detector," in Proc. ECCV, 2016, pp. 21–37.

[5] J. Li, Y. Pan, and M. Zhang, "Real-Time Pedestrian Detection in Crowded Scenes Using YOLOv5," J. Real-Time Image Process., vol. 19, no. 4, pp. 823–835, Aug. 2022.

[6] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," in Proc. ECCV, 2020, pp. 213–229.

[7] G. Jocher et al., "Ultralytics YOLOv8," GitHub repository, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[8] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information," in Proc. ECCV, 2024, pp. 1–21.

[9] J. J. Fruin, Pedestrian Planning and Design. New York, NY, USA: Metropolitan Association of Urban Designers and Environmental Planners, 1971.

[10] D. Helbing and P. Molnar, "Social Force Model for Pedestrian Dynamics," Phys. Rev. E, vol. 51, no. 5, pp. 4282–4286, May 1995.

[11] A. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv5," GitHub repository, 2020. [Online]. Available: https://github.com/ultralytics/yolov5

[12] T.-Y. Lin et al., "Feature Pyramid Networks for Object Detection," in Proc. IEEE CVPR, 2017, pp. 2117–2125.

[13] M. Abdulla, "Roboflow: Computer Vision Tools for Developers," Roboflow, Inc., 2020. [Online]. Available: https://roboflow.com

[14] F. Zhu, C. Li, W. Ouyang, X. Wang, and X. Bai, "Multi-Scale Head Detection in Dense Crowd Scenes," IEEE Trans. Image Process., vol. 31, pp. 3435–3447, 2022.

[15] X. Cao, Z. Wang, Y. Zhao, and F. Su, "Scale Aggregation Network for Accurate and Efficient Crowd Counting," in Proc. ECCV, 2018, pp. 757–773.