International Journal on Advanced Science Engineering Information Technology

Vol.15 (2025) No. 1
ISSN: 2088-5334

Human Detection System Using Machine Learning to Calculate Crowd Potential

Eni Dwi Wardihani [a,*], Rindang Ayu Oktaviani [a], Ricky Sambora [a], Eko Supriyanto [a], Amin Suharjono [a], Rizkha Ajeng Rochmatika [a], Catur Budi Waluyo [a], Muhlasah Novitasari Mara [a], Ari Sriyanto Nugroho [a], Aminuddin Rizal [b], Suko Tyas Pernanda [b]

[a] *Department of Telecommunication Engineering, Politeknik Negeri Semarang, Tembalang, Semarang, Indonesia*
[b] *Department of Electronics engineering, Politeknik Negeri Semarang, Tembalang, Semarang, Indonesia*
Corresponding author: [*]edwardihani@polines.ac.id

***Abstract*—Crowds are a common social phenomenon occurring in various settings such as large gatherings, public transportation, and popular tourist attractions. Crowding poses significant risks as it can lead to scenarios known as human stampedes. In incidents such as those at Kanjuruhan and Itaewon, the presence of large crowds caused individuals to lose their footing, resulting in falls, trampling, and respiratory complications. This issue is further exacerbated by current crowd detection techniques, which still rely on manual observation. To address this problem, a machine learning-based system was developed for human detection by counting the number of detected heads to evaluate crowd capacity. This study employs a combination of Convolutional Neural Network (CNN) architecture and the YOLOv8 algorithm, trained on a custom dataset and implemented on Jetson Nano for real-time monitoring via a website hosted on localhost. The dataset was created by collecting images of crowded locations, such as bus stops during peak commuting hours and shopping malls on weekends. Testing was conducted at Kantik Kolam Berkah in Politeknik Negeri Semarang during lunch hours on weekdays. Density estimation was performed by calculating the number of detected heads divided by the area being observed, yielding a density figure for the area. The findings reveal that the developed system can identify individuals and measure density with an average precision of 0.965 and an average recall of 0.765, with an average inference time of 283.73 milliseconds (ms).**

***Keywords*—Crowd; Jetson Nano; YOLOv8; real-time.**

Manuscript received 7 Aug. 2024; revised 17 Oct. 2024; accepted 11 Nov. 2024. Date of publication 28 Feb. 2025.

I. INTRODUCTION

A crowd can be defined as a group of pedestrians gathered in a particular place for similar or sometimes different purposes. Crowds are present in most places and have become an integral part of daily life [1]. Overcrowded situations can pose a serious danger as they may lead to incidents known as human stampedes, where people are pushed against each other while moving, potentially causing injuries or even fatalities.

Traditional crowd detection methods have evolved from manually crafted techniques to approaches using machine learning and artificial intelligence. These traditional methods are unsuitable for modern surveillance due to their low accuracy and the dynamic nature of current crowd conditions. Consequently, machine learning techniques have been adopted for more effective crowd analysis. Previous studies [2] have employed Raspberry Pi 4 modules as microcomputers. However, the application of machine learning on Raspberry Pi 4 modules has been less effective, with very slow detection speeds due to reliance on the Central Processing Unit (CPU) for data processing. Moreover, the benchmark achieved by Raspberry Pi 4 is 8.1 frames per second (fps) [3].

Object detection is a technique in the field of computer vision that focuses on recognizing and locating objects within an image or video recording. This technique utilizes machine learning or deep learning methods to achieve accurate detection. Its presence is crucial in various sectors such as security monitoring, autonomous vehicles, and visual media analysis, as it enables machines to identify and classify objects with high effectiveness.

In neural networks, basic features that already exist are used repeatedly to create more complex features. This approach is known as transfer learning, which allows leveraging existing knowledge to enhance performance in different tasks. Transfer learning is a technique in deep

learning where a pre-trained model is used as a starting point or base to train a new model on a different task. By harnessing knowledge learned from a given task, a new model can learn faster or with a smaller dataset. This approach is particularly effective when the available dataset for training a new model is limited or when the task to be solved shares characteristics with the task the model was originally trained on.

The research conducted explored several object detection models, each exhibiting distinct advantages and limitations. Notably, the Faster R-CNN model is widely recognized for its high precision, making it the preferred choice in contexts where accuracy is paramount. As highlighted in studies [4], [5], and [6], Faster R-CNN consistently demonstrates superior accuracy when compared to YOLO models. However, its slower inference speed renders it less ideal for real-time applications, which are critical in crowd detection systems designed to mitigate risks in high-density environments.

Conversely, the YOLO family of models presents a favorable compromise between speed and accuracy, positioning them as more suitable for real-time detection tasks. Among these, comparative analyses of YOLOv3, YOLOv5, and MobileNet-SSD V2 indicate that YOLOv5 offers the optimal balance of speed and accuracy [7]. Moreover, the recent introduction of YOLOv8 has further enhanced precision, showing a 2.82% improvement over YOLOv5 [8]. YOLOv8's enhanced architecture, which facilitates faster inference, renders it particularly well-suited for real-time monitoring on resource-constrained devices such as the Jetson Nano.

The selection of YOLOv8 for this study is informed by its capacity to maintain a high level of accuracy while achieving the necessary inference speed for real-time crowd detection. This balance between speed and accuracy is crucial for ensuring timely interventions in dynamic environments. Furthermore, YOLOv8's streamlined architecture allows for more efficient utilization of Jetson Nano's limited computational resources, ensuring stable performance even under varying environmental conditions.

II. Materials and Method

A. Crowd

A crowd consists of individuals who spontaneously gather in a certain area at the same time. These individuals congregate temporarily and disperse when they separate [11]. A crowd does not have a single mutually agreed-upon purpose. According to the Green Guide, published by the British government, the maximum allowable density is 4.7 people per square meter (m^2). The threshold for potentially dangerous dynamic crowd density is 5.55 people per m^2, and the critical density limit for static crowds is 7.1 people per m^2 [9].

$$\text{Density} = \frac{The\ number\ of\ people}{Total\ area\ (m^2)} \quad (1)$$

B. YOLOv8

YOLO stands for "You Only Look Once", which is one of the popular approaches in object detection in the field of computer vision. This method is designed to identify and localize objects in images or videos quickly and accurately. YOLO works by dividing the image into a grid and generating a prediction for each grid cell, including the probability of object presence and the bounding box surrounding the object. This method is known for its ability to solve real-time object detection problems at high speed, making it suitable for applications such as security surveillance, autonomous vehicles, and object recognition in video.

The YOLOv8 architecture is organized into three primary components: the Backbone, Neck, and Head. The Backbone, based on CSPDarknet53, is responsible for extracting features from the input image, serving as a robust foundation for subsequent processing stages. The Neck, utilizing the C2f module, fuses feature maps from various layers, enabling the capture of both high-level semantic information and low-level spatial details, which is particularly beneficial for detecting smaller objects. Lastly, the Head performs predictions by generating bounding boxes and object classifications for each detected object.

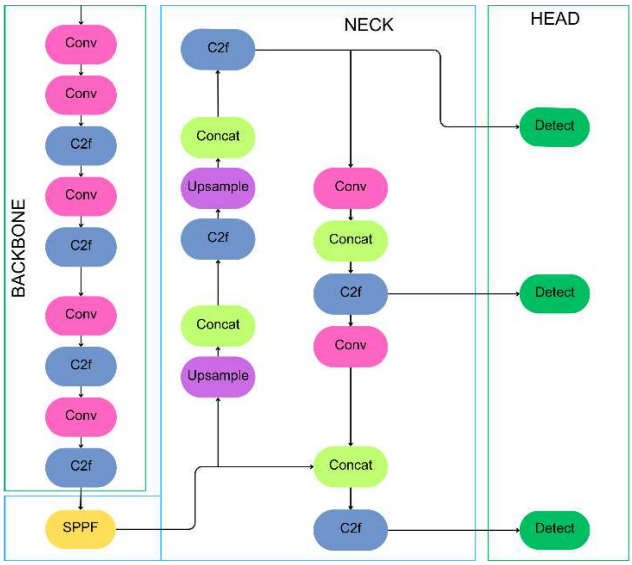

Fig. 1 YOLOv8 Architecture

C. Architecture

The architecture of the human crowd detection system using Jetson Nano as an edge device consists of three main components: the camera, the edge device, and the website. The camera is used to capture images or videos of the surrounding environment. These images or videos are then sent to the edge device for processing. The edge device utilizes crowd detection algorithms to process the images or videos and determine if there is a crowd in the area. The crowd detection results are then sent to the website for display.

The training process involves creating a new model suitable for detecting crowds based on the YOLOv8 model. Transfer learning is performed using Google Collaboratory with YOLOv8 and the Ultralytics library and then converted into TensorRT format. The dataset used includes a private dataset consisting of 11,169 training data, 2,394 validation data, and 2,394 test data. The initial step is labeling the dataset and creating a label map for the classes. The dataset is then converted into a format suitable for YOLO training. During the training process, it is necessary to select the model to be used, namely YOLOv8, and configure the parameters for epochs and batch size for training. The final step is converting the model into TensorRT format and evaluating the model. The parameters used for evaluation include the model's

precision and recall, as well as calculating the Mean Average Precision (mAP) and the confusion matrix. The final results are then sent to the web server using the Flask library for streaming display.

D. System

The input from the camera is processed to count the number of visible people. Once the count is determined, the system will assess the situation based on the number of people. If the number of detected people is less than 25% of the maximum allowable number, a green indicator will appear. If it is more than 25%, the number of detected people will be reassessed. If it is less than 50%, a yellow indicator will appear. If it is more than 50%, the number of detected people will be reassessed again. If it is less than 75%, an orange indicator will appear. If it is more than 75%, a red indicator will appear. After determining the appropriate indicator, the result will be displayed on the website.

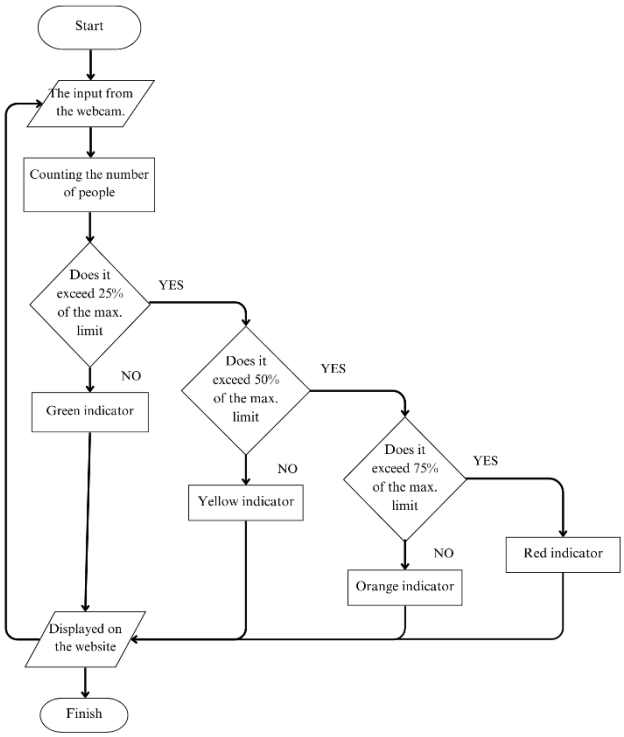

Fig. 2 System principles

E. Dataset

The dataset used is a private dataset. This dataset was created by collecting photos from various places in Semarang City, Indonesia. In particular, the photos taken are photos of places that have high density such as bus stops during peak hours such as going to and from work and malls on weekends.

These images are then annotated with a label indicating the object, which in this case is a human head, a process known as labelling with Roboflow. The dataset used is ensured to be in the appropriate format for training the YOLO model. The dataset consists of 15,957 images, divided into 70% training data, 15% validation data, and 15% test data.

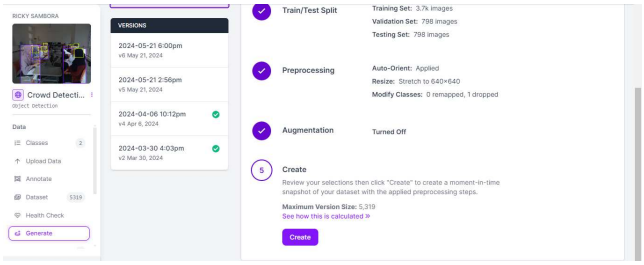

Fig. 3 Roboflow

F. Training

The model training process is conducted using Google's facility known as Google Collaboratory, which requires a Google account to initiate the training process. The initial step of the model training process involves uploading the prepared dataset and selecting the base model to be utilized. Subsequently, training parameters are configured according to requirements, and the training process commences. This training phase may take several hours to several days to complete. Following the model training, an evaluation is performed to assess the performance and effectiveness of the trained model. The trained model's results are then converted into an engine format, which is subsequently deployed onto Jetson Nano for inference purposes.

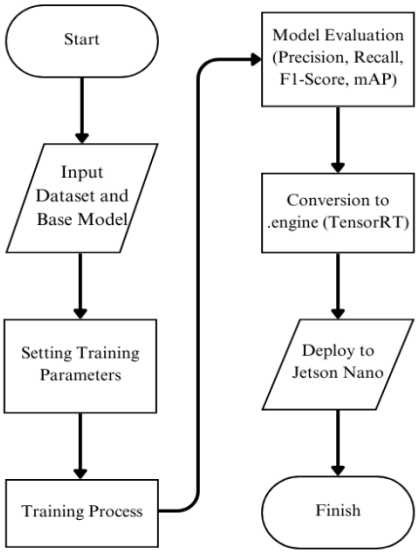

Fig. 4 Training Process

G. Inference

Inference is the process of using a trained model to make predictions based on previously unseen data. The process begins by inputting the model to be used. Subsequently, input from the camera is provided. OpenCV detects the presence of frames. If no frames are detected, the process terminates. If frames are detected, the process continues with YOLO for human counting. After humans are counted, head annotation is performed by Supervision. Then, the frames that are ready are recombined by OpenCV.

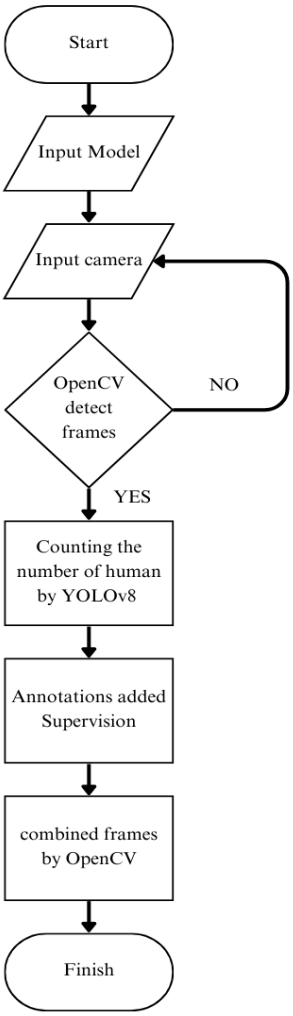

Fig. 5 Inferences Process

H. Web

A mock-up is an initial visual planning of the appearance and layout of a website. On the main page of the website, there is a connected Wi-Fi SSID, device IP, and internet connection conditions. On the main page, there are also inference parameters, the IoU set, the confidence used, and the maximum population in the zone. Then, on the "Edit" menu, it will be used as a page to change the parameters so that they can be adjusted according to the needs.

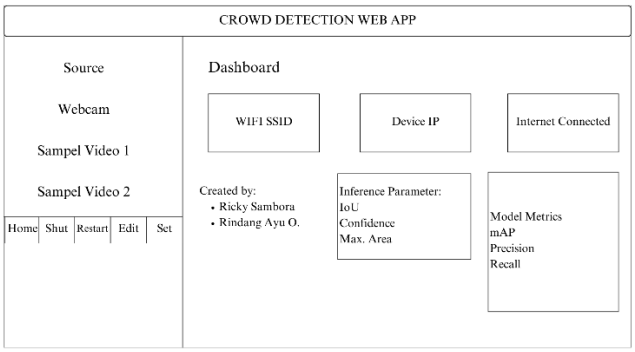

Fig. 6 Web Mock-up

I. Testing

The test was conducted for 2 days during the lunch break at the Semarang State Polytechnic Pool Canteen with a floor area of 163.5 m². The confidence level used is 0.45. Testing was carried out during the cafeteria's peak hours, namely during lunchtime. Testing begins with the preparation of tools and systems to be tested. Next, the system is activated, and input is received from the camera. The inference process is performed every 5 frames and is immediately displayed on the website. The final result of this test is a video recording that will be used for precision and recall analysis.

Precision is used to measure the accuracy of positive prediction results, while recall is used to measure the sensitivity of the developed model.

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

III. Results and Discussion

A. Model Evaluation

Based on the model evaluation results, the precision obtained is 0.95 and the recall is 0.87, with a Mean Average Precision (mAP) of 0.93.

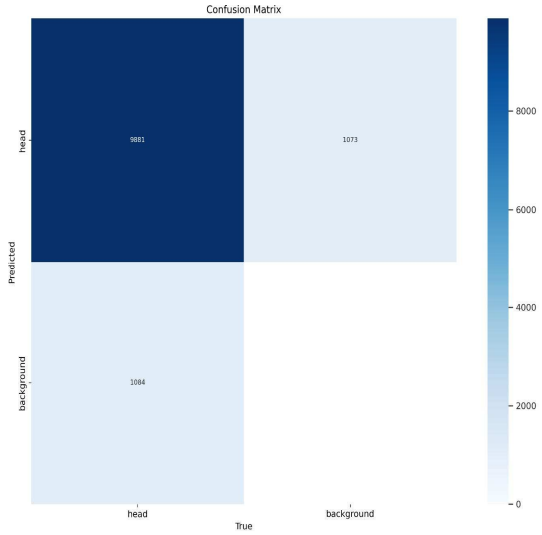

Fig. 7 Confusion Box of Evaluation Model Result

B. Detection Result

As an example of the detection results, it is observed that 32 heads are detected with a green annotation. The green annotation indicator signifies that the detected density is less than 25% of the recommended maximum density limit of 4.7 people/m².

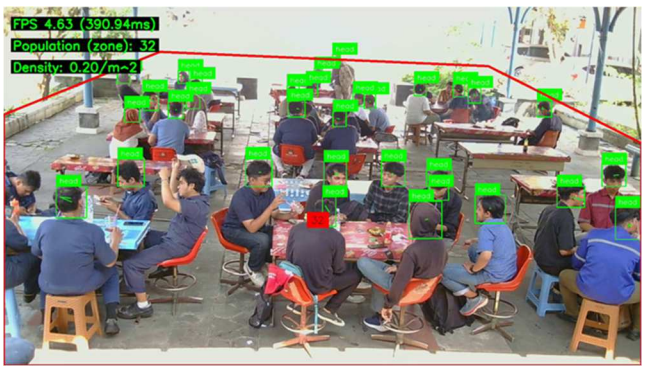

Fig. 8 Detection Result

C. *Testing Result*

The testing was conducted at the Berkah Kolam Cafeteria of the State Polytechnic of Semarang with a floor area of 163.5 m². A confidence level of 0.45 was utilized. Over the course of 2 days, a total of 800 images were obtained from the test. In total, over the course of 2 days of testing, there were 19,877 True Positives, 744 False Positives, and 5,809 False Negatives obtained.

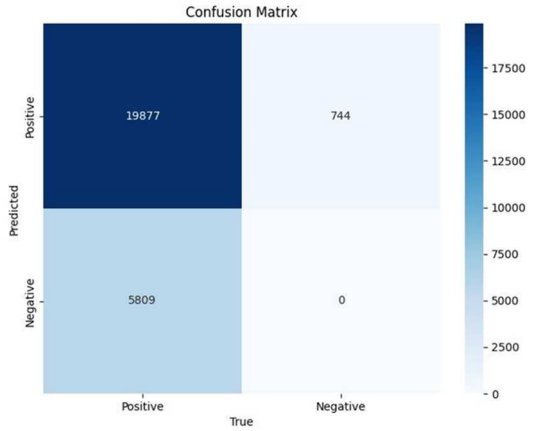

Fig. 9 Confusion Box of Testing Result

In the model evaluation using the test dataset, a precision of 0.95 and a recall of 0.87 were obtained. These high evaluation scores indicate that the model has been well-trained and can detect head objects under conditions similar to those of the training data. However, different dataset conditions can yield varying results in testing.

Based on the testing conducted at the Berkah Kolam Cafeteria over 2 days, a precision of 0.965 and a recall of 0.765 were achieved. The difference in precision between field testing and evaluation is not significant, with a 0.015 higher result observed in the field testing. However, there is a more notable difference in recall, which is 0.105 lower compared to the evaluation results.

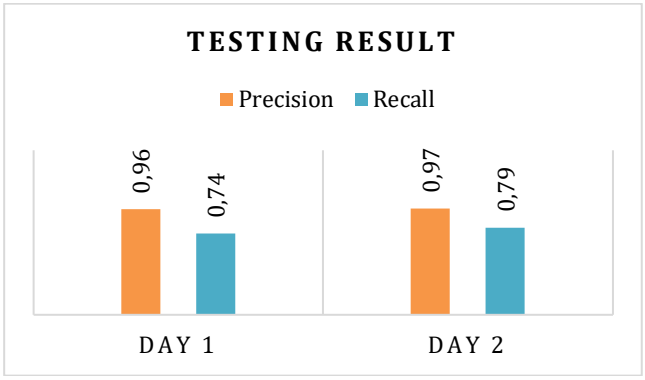

Fig. 10 Difference of Testing Result

These results indicate that the model still has limitations in detecting human heads. This constraint is primarily caused by several factors during testing, notably the overlapping positions of human heads within the detection area. Consequently, the captured head area in the processed images is very small. In such cases, the features or characteristics received by the neural network model may inadequately represent the shape of heads, resulting in low confidence scores or undetected instances. Enhancing the input resolution of the image to be detected and reducing the IoU value aims to improve the accuracy of detecting densely populated human areas. However, this approach can only be implemented on edge devices with higher computational capacity and more complex models.

It is important to emphasize that the confidence threshold used to capture data in Table 4.1 is set at 0.45. Therefore, if the confidence score falls below this threshold, bounding box detection results will not appear.

TABLE I
TESTING RESULT

Day	TP	FP	FN	Avg. Inference Time
1	7420	306	2571	289,26 (ms)
2	12457	438	3238	278,20 (ms)

The method used for inference involves splitting the image into 4 tiles to predict separately and detect small or distant objects and predicting every 5 frames to reduce device load. Both methods contribute to increased false positives and false negatives in detection. The technique of dividing the image into 4 tiles helps improve detection results for small and distant objects from the camera. However, this also leads to occurrences of false negatives and duplicate detections. This happens because objects may lie at the intersection of tiles, causing only partial detection of features or characteristics, which limits optimal detection. In specific cases, this may result in two bounding boxes appearing for one object because the head is divided exactly at the tile intersection. In other conditions, a bounding box may only appear on one tile due to differences in the proportion of the object segment, which tends to fall within one tile. Consequently, after merging tiles into a single image, the bounding box may only partially cover the head area. Lastly, insufficient feature extraction from object segments in each tile can result in confidence scores falling below the threshold, leading to detections being considered non-existent.

Fig. 11 Tile Division

The occurrence of false positives is also influenced by the prediction method or inference every 5 frames. This means that predictions are only run once on the initial image and applied to the next 4 frames. This process reduces the device load, as processing one image takes a long time, averaging 283.73 ms. A drawback of this method is demonstrated by false positives when detecting moving objects. The resulting bounding boxes lag behind and do not follow the object,

necessitating a reduction in the detection distance between frames for using this method in situations involving moving objects.

The average inference time obtained when processing a single image is 283.73 ms. This result is influenced by dividing the image into 4 tiles. Predicting at a resolution of 640x640 takes varying times, typically ranging from 60 ms to 80 ms. Therefore, the time required to process an image with 4 tiles is between 240 ms and 320 ms. This factor is what causes the inference process to take longer.

The deployment of the crowd detection system on the Jetson Nano encountered several challenges due to the device's hardware limitations. Although the Jetson Nano is equipped with a CUDA-capable GPU, its older architecture resulted in low fps performance when using the YOLOv8 nano model, even after reducing the number of annotated frames. To enhance performance, the smallest variant of the model was selected, and TensorRT integration was employed, which improved inference speed but remained limited by the device's processing capacity.

While the system's average inference time is sufficient for real-time applications using a single video feed, the processing of high-resolution images or multiple camera feeds led to significantly slower inference times, rendering it impractical for scenarios requiring the simultaneous analysis of multiple video streams.

IV. Conclusion

The human detection system leveraging machine learning has been evaluated, demonstrating an average precision rate of 0.965, indicating that 96.5% of instances classified as crowds were accurate. It also achieved a recall rate of 0.765, successfully detecting 76.5% of actual crowd occurrences. With an inference time of approximately 283.73 milliseconds, the system maintains a response time that, while not ideal for real-time applications, is suitable for monitoring environments with moderate crowd densities.

However, the system encounters challenges related to scalability and robustness. Jetson Nano's limited processing power restricts its ability to efficiently process multiple camera feeds or high-resolution images. Furthermore, the system's accuracy can fluctuate under conditions such as occlusions or suboptimal camera angles, suggesting the need for additional training with more diverse datasets. Moreover, real-world deployment of such systems must address ethical concerns, particularly regarding privacy, to ensure responsible use.

References

[1] S. A. H. Al-Gadhi, "A Review Study of Crowd Behavior and Movement," *J. King Saud Univ. - Eng. Sci.*, vol. 8, no. 1, pp. 77–107, 1996, doi: 10.1016/S1018-3639(18)30641-X.

[2] A. Fadlil and D. Prayogi, "Face Recognition Using Machine Learning Algorithm Based on Raspberry Pi 4b," *Int. J. Artif. Intell. Res.*, vol. ISSN, no. 1, pp. 2579–7298, 2022, doi: 10.29099/ijair.v7i1.321.

[3] M. Luqman Bukhori and E. E. Prasetiyo. (2023). "Sistem Deteksi Masker Berbasis Jetson Nano dengan Deep Learning Framework TensorFlow". *Jurnal Nasional Teknik Elektro Dan Teknologi Informasi*, *12*(1), 15-21. doi: 10.22146/jnteti.v12i1.5472

[4] L. Ezzeddini *et al.*, "Analysis of the performance of Faster R-CNN and YOLOv8 in detecting fishing vessels and fishes in real time," *PeerJ Comput. Sci.*, vol. 10, p. e2033, 2024, doi: 10.7717/peerj-cs.2033.

[5] I. Karamouzas, N. Sohre, R. Hu, and S. J. Guy, "Crowd space: A predictive crowd analysis technique," *SIGGRAPH Asia 2018 Tech. Pap. SIGGRAPH Asia 2018*, vol. 37, no. 6, 2018, doi:10.1145/3272127.3275079.

[6] R. Bai, F. Shen, M. Wang, J. Lu, and Z. Zhang, "Improving Detection Capabilities of YOLOv8-n for Small Objects in Remote Sensing Imagery: Towards Better Precision with Simpliied Model Complexity Improving Detection Capabilities of YOLOv8-n for Small Objects in Remote Sensing Imagery: Towards Better Pre," *Res. Sq.*, pp. 0–9, 2023. doi: 10.21203/rs.3.rs-3085871/v1

[7] R. Iyer, P. Shashikant Ringe, R. Varadharajan Iyer, and K. Prabhulal Bhensdadiya, "Comparison of YOLOv3, YOLOv5s and MobileNet-SSD V2 for Real-Time Mask Detection," *Artic. Int. J. Res. Eng. Technol.*, vol. 8, no. 7, pp. 1156–1160, 2021.

[8] I. P. Sary, S. Andromeda, and E. U. Armin, "Performance Comparison of YOLOv5 and YOLOv8 Architectures in Human Detection using Aerial Images," *Ultim. Comput. J. Sist. Komput.*, vol. 15, no. 1, pp. 8–13, 2023, doi: 10.31937/sk.v15i1.3204.

[9] M. Wirz, T. Franke, D. Roggen, E. Mitleton-Kelly, P. Lukowicz, and G. Tröster, "Probing crowd density through smartphones in city-scale mass gatherings," *EPJ Data Sci.*, vol. 2, no. 1, pp. 1–24, 2013, doi:10.1140/epjds17.

[10] N. N. Amir Sjarif, S. M. Shamsuddin, S. Z. Mohd Hashim, and S. S. Yuhaniz, "Crowd analysis and its applications," *Commun. Comput. Inf. Sci.*, vol. 179 CCIS, no. PART 1, pp. 687–697, 2011, doi:10.1007/978-3-642-22170-5_59.

[11] J. C. S. Jacques, S. R. Mussef, and C. R. Jung, "Crowd analysis using computer vision techniques," *IEEE Signal Process. Mag.*, vol. 27, no. 5, pp. 66–77, 2010, doi: 10.1109/MSP.2010.937394.

[12] H. Yan, "Deadly crowd surges have happened for decades. Safety standards exist, but they're not required nationwide," CNN, Nov. 12, 2021. https://edition.cnn.com/2021/11/11/us/safety-standards-requirements-crowd-surges/index.html

[13] Abdulkadir Gozuoglu, Okan Ozgonenel, and Cenk Gezegin, "CNN-LSTM Based Deep Learning Application on Jetson Nano: Estimating Electrical Energy Consumption for Future Smart Homes," *Internet of things*, vol. 26, pp. 101148–101148, Jul. 2024, doi:10.1016/j.iot.2024.101148.

[14] V. Gonzalez-Huitron, J. A. León-Borges, A. E. Rodriguez-Mata, L. E. Amabilis-Sosa, B. Ramírez-Pereda, and H. Rodriguez, "Disease detection in tomato leaves via CNN with lightweight architectures implemented in Raspberry Pi 4," *Computers and Electronics in Agriculture*, vol. 181, p. 105951, Feb. 2021, doi:10.1016/j.compag.2020.105951.

[15] Y. Chen, Y.-F. Li, C. Cheng, and H. Ying, "Neural network based cognitive approaches from face perception with human performance benchmark," *Pattern recognition letters*, Jun. 2024, doi:10.1016/j.patrec.2024.06.024.

[16] J. Yan *et al.*, "Enhanced object detection in pediatric bronchoscopy images using YOLO-based algorithms with CBAM attention mechanism," *Heliyon*, vol. 10, no. 12, pp. e32678–e32678, Jun. 2024, doi: 10.1016/j.heliyon.2024.e32678.

[17] Sushila Palwe, A. Gunjal, S. Jindal, A. Shrivastava, A. Deshmukh, and Mehul Navalakha, "An Intelligent and Deep Learning Approach for Pothole Surveillance Smart Application," *Procedia computer science*, vol. 235, pp. 3271–3282, Jan. 2024, doi: 10.1016/j.procs.2024.04.309.

[18] E. Casas, L. Ramos, C. Romero, and Francklin Rivas-Echeverría, "A comparative study of YOLOv5 and YOLOv8 for corrosion segmentation tasks in metal surfaces," *Array*, vol. 22, pp. 100351–100351, Jul. 2024, doi: 10.1016/j.array.2024.100351.

[19] F. Hou, W. Lei, S. Li, J. Xi, M. Xu, and J. Luo, "Improved Mask R-CNN with distance guided intersection over union for GPR signature detection and segmentation," *Automation in Construction*, vol. 121, p. 103414, Jan. 2021, doi: 10.1016/j.autcon.2020.103414.

[20] X. Zhang *et al.*, "Area in circle: A novel evaluation metric for object detection," *Knowledge-based systems*, vol. 293, pp. 111684–111684, Jun. 2024, doi: 10.1016/j.knosys.2024.111684.

[21] Mohd Nazuan Wagimin *et al.*, "Classification model for chlorophyll content using CNN and aerial images," *Computers and electronics in agriculture*, vol. 221, pp. 109006–109006, Jun. 2024, doi:10.1016/j.compag.2024.109006.

[22] X. Yang, Y. Liu, A. Majumdar, E. Grass, and W. Ochieng, "Characteristics of crowd disaster: database construction and pattern identification," International journal of disaster risk reduction, pp. 104653–104653, Jul. 2024, doi: 10.1016/j.ijdrr.2024.104653.

[23] Z. Zhu and Y. Cheng, "Application of attitude tracking algorithm for face recognition based on OpenCV in the intelligent door

lock," *Computer Communications*, vol. 154, pp. 390–397, Mar. 2020, doi: 10.1016/j.comcom.2020.02.003.

[24] D. Syrlybayev, N. Nauryz, A. Seisekulova, K. Yerzhanov, and Md. H. Ali, "Smart Door for COVID Restricted Areas," Procedia Computer Science, vol. 201, pp. 478–486, Jan. 2022, doi:10.1016/j.procs.2022.03.062.

[25] T. Peng-o and P. Chaikan, "High performance and energy efficient sobel edge detection," Microprocessors and Microsystems, vol. 87, p. 104368, Nov. 2021, doi: 10.1016/j.micpro.2021.104368.

[26] J. Lee, M. Yu, Y. Kwon, and T. Kim, "Quantune: Post-training quantization of convolutional neural networks using extreme gradient boosting for fast deployment," Future Generation Computer Systems, vol. 132, pp. 124–135, Jul. 2022, doi: 10.1016/j.future.2022.02.005.

[27] Islomjon Shukhratov, Andrey Pimenov, A. Stepanov, Nadezhda Mikhailova, A. Baldycheva, and Andrey Somov, "Optical Detection of Plastic Waste Through Computer Vision," Intelligent systems with applications, pp. 200341–200341, Feb. 2024, doi:10.1016/j.iswa.2024.200341.

[28] J. Amin, Irum Shazadi, M. Sharif, M. Yasmin, Nouf Abdullah Almujally, and Y. Nam, "Localization and Grading of NPDR Lesions Using ResNet-18-YOLOv8 Model and Informative Features Selection for DR Classification based on Transfer Learning," Heliyon, pp. e30954–e30954, May 2024, doi: 10.1016/j.heliyon.2024.e30954.

[29] B. Ganga, Lata B.T, and Venugopal K.R, "Object detection and crowd analysis using deep learning techniques: Comprehensive review and future directions," Neurocomputing, pp. 127932–127932, May 2024, doi: 10.1016/j.neucom.2024.127932.

[30] Y. C. Putra and A. W. Wijayanto, "Automatic detection and counting of oil palm trees using remote sensing and object-based deep learning," Remote Sensing Applications: Society and Environment, vol. 29, p. 100914, Jan. 2023, doi: 10.1016/j.rsase.2022.100914.

[31] A. Deshpande and K. Warhade, "SADY: Student Activity Detection Using YOLO-based Deep Learning Approach," International Journal on Advanced Science, Engineering and Information Technology/International journal of advanced science, engineering and information technology, vol. 13, no. 4, pp. 1501–1501, Jul. 2023, doi: 10.18517/ijaseit.13.4.18393.