JAIIT *(Journal of Advances in Information and Industrial Technology)*
Vol. 7, No. 2 Nov 2025, ISSN 2716-1935, E-ISSN 2716-1927

Implementation of the YOLOv8n Model for Automatic Owl Detection in Swiftlet Farming Buildings

Iqbal Kurniawan Asmar Putra*1), Apriska Prameswari2), Muhammad Ainul Fikri3), Ahmad Riznandi Suhari4).
1,2) Electrical Engineering, Politeknik Negeri Padang, Padang, Indonesia
3) Informatics Engineering, Politeknik Negeri Jember, Jember, Indonesia
4) Information Science and Technology, Tokai University, Tokyo, Japan
1*iqbalkurniawan@pnp.ac.id, 2apriska@pnp.ac.id, 3m.ainulfikri@polije.ac.id, 45MTAD004@tokai.ac.jp

ARTICLE INFO

Article history:
Received 31 October 2025
Revised 5 November 2025
Accepted 26 November 2025
Available online 29 November 2025

Keywords:
Deep Learning
Object Detection
Owl
YOLOv8
YOLOv8n

ABSTRACT

Object detection based on digital images is a rapidly developing field in the application of intelligent systems. This study aims to create an automatic owl detection system utilizing the YOLOv8 deep learning model as a pest mitigation measure in the swiftlet farming industry. Owls are known to enter swiftlet houses at night and prey on the birds, causing economic losses. Owl image datasets were obtained from the Roboflow platform and annotated in YOLO format. The model was trained using the YOLOv8-nano architecture with a 640×640 pixel input resolution. The evaluation results showed that the model achieved a mAP@0.5 of 96.82% and mAP@0.5:0.95 of 70.5%, with a precision of 97.2% and a recall of 93.38%. These results indicate that the YOLOv8 model performs well and has the potential to be implemented as an automatic monitoring system in swiftlet farming environments.

1. Introduction

The integration of artificial intelligence (AI) into agricultural and livestock farming sectors has significantly increased in recent years, particularly in developing countries such as Indonesia. This growth is driven by the need for more efficient environmental monitoring, pest control, and production management systems. The adoption of intelligent systems based on computer vision has become a promising solution to address these challenges. As practical field issues such as wildlife intrusion and environmental instability demand more effective interventions, the development of automated surveillance technologies has emerged as a crucial innovation to improve farming productivity and sustainability [1].

The swiftlet farming industry represents a high-value economic sector in Indonesia, with its primary product, edible bird's nests, commanding premium prices in both domestic and international markets. This sustained market demand has catalyzed the proliferation of swiftlet farming facilities across various regions [2]. Despite its economic importance, the operational management of swiftlet farming remains challenging, particularly in controlling environmental conditions and preventing disturbances from predatory species such as owls. These predators frequently enter swiftlet buildings at night to prey on the birds, causing direct population losses, damaging nest structures, and reducing the overall yield.

Beyond direct economic damage, owl intrusions also affect the behavioral and physiological conditions of swiftlets. Prolonged exposure to predators induces stress, leading to disrupted breeding patterns and decreased nest production rates. As a result, many farmers have reported a decline in productivity due to nocturnal disturbances. Traditional monitoring methods, such as manual inspections or static deterrent devices, have proven ineffective, especially in large or multi-floor facilities.

Therefore, developing an automated detection and early-warning system is essential to protect the swiftlet population and ensure sustainable farming operations [3].

Recent advancements in digital image processing and artificial intelligence have created significant opportunities for developing automated detection systems [4]. Among the various approaches in object detection, deep learning methods have gained considerable traction due to their capability to automatically recognize and classify objects within images or video streams [3]. These technologies have found widespread application across multiple sectors, including security, agriculture, transportation, and manufacturing industries [5].

Within the context of livestock farming environmental monitoring, the implementation of deep learning-based object detection methods can significantly enhance surveillance efficiency and reduce reliance on manual oversight. Among the prominent algorithms employed for object detection tasks is You Only Look Once (YOLO) [6], [7], [8], [9], [10]. YOLO is a single-stage detection method that offers high-speed processing and adequate accuracy for real-time applications. The latest iteration of this algorithm, YOLOv8n [11], [12], has undergone substantial refinements in architecture, processing efficiency, and detection quality [8][10].

This study aims to implement YOLOv8n in an automated owl detection system as part of a swiftlet farm monitoring framework [7]. The system is designed to identify the presence of owls quickly and accurately from image data captured in real conditions, enabling early intervention and damage prevention. The contributions of this research are threefold: (1) the development of a YOLOv8n-based detection model optimized for owl detection; (2) a comprehensive performance evaluation of the model on a specialized dataset; and (3) an analysis of its deployment feasibility within real-time monitoring systems for swiftlet farming applications.

2. Methods

This research was conducted through several key stages: dataset collection and preparation, model training using YOLOv8n, and performance evaluation on test data.

a. Dataset and Annotation

The dataset used in this study consists of 1,038 annotated owl images, obtained from the Roboflow platform and labeled in YOLO format. The dataset was divided into 70% training (727 images), 20% validation (207 images), and 10% testing (104 images) to ensure a balanced evaluation workflow. The images exhibit diverse visual characteristics, diverse background compositions, and multiple owl orientations and distances. These variations reflect realistic conditions encountered inside swiftlet farming buildings and are essential for improving the model's robustness in real-world deployment [13].

b. Model Architecture YOLOv8n

YOLOv8n (nano)[14] represents the lightweight variant within the YOLOv8 family, engineered for high efficiency with a compact model size while maintaining competent detection accuracy. This architecture is designed for deployment on devices with limited computational resources, such as edge devices or cloud-based monitoring systems with GPU constraints. A key advantage of YOLOv8n lies in its utilization of C2f layers and a single-stage detector structure, which accelerates the inference process without significantly compromising accuracy [15].

The YOLOv8n architecture comprises three primary components: the Backbone, Neck, and Output. Figure 1 illustrates the network structure of YOLOv8n employed in this study. Within the Backbone, feature extraction is conducted through multiple Conv and C2f layers, responsible for deriving spatial features from the input image. The SPPF (Spatial Pyramid Pooling Fast) module functions to capture information across various scales [16].

Subsequently, the Neck section amalgamates features from different scales using Concat, Upsample, and C2f layers. This process aims to integrate information from multiple resolutions to enhance detection capability for objects of varying sizes [17], [18].

 DOI: https://doi.org/10.52435/jaiit.v7i2.733

The final component is the Output, which generates object detection predictions. In the YOLOv8n-seg structure, the output is adapted to support segmentation tasks; however, within the context of this research, it is specifically utilized for owl detection [16].

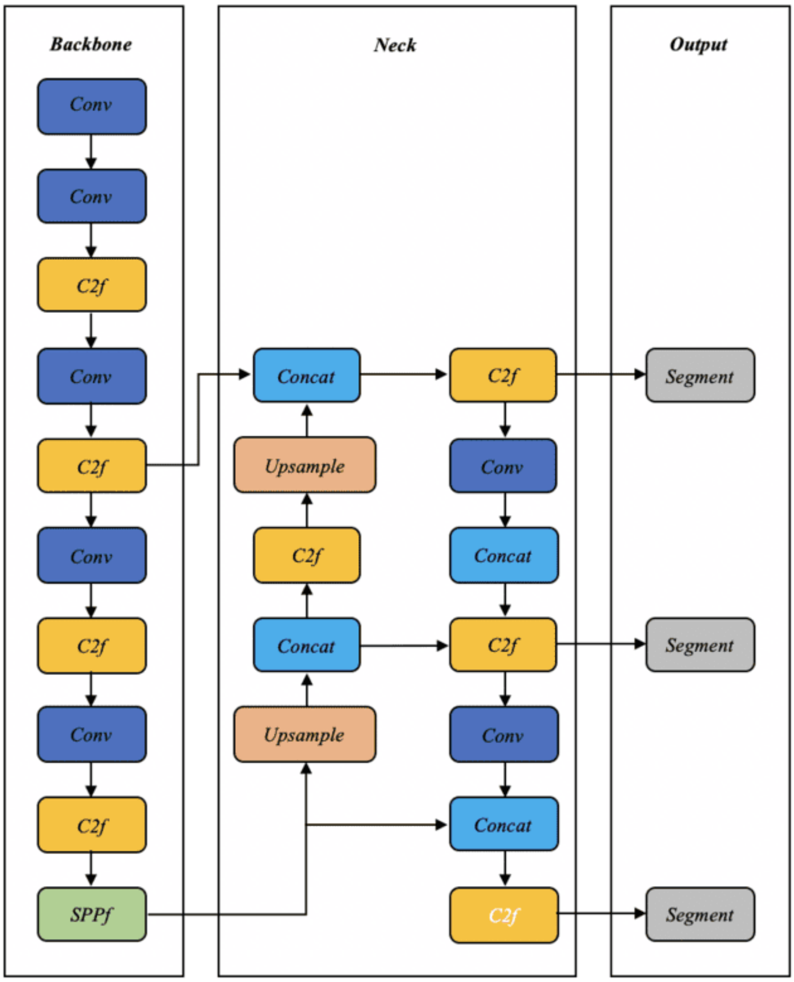

Figure 1. YOLOv8n Structure [19]

The model employed in this study is YOLOv8n (nano), the smallest variant within the YOLOv8 family, characterized by a lightweight parameter count that ensures efficiency for training on resource-constrained devices. YOLOv8n is a single-stage detection model that integrates feature extraction, regression, and classification within a unified, compact architecture. This model demonstrates the capability to detect objects across various scales with both high speed and accuracy [20].

c. Software Specifications

The Roboflow platform was selected for its user-friendly data annotation tools and seamless integration with the YOLOv8 framework via API. Additionally, Google Colab provides complimentary GPU access, which significantly facilitates the deep learning model training process [21], [22]. The training environment employed in this study encompasses several integrated hardware and software components to support model development. Table 1 presents the primary tools utilized throughout the research implementation.

The training process was conducted using Google Colab with complimentary GPU support. Visual Studio Code was used for writing the program code employed in the study [23]. The main library utilized is Ultralytics YOLOv8, installed via pip. The dataset was imported using the Roboflow API, connected directly to the training project [13].

Table 1. Hardware and Platforms Used in the Research

No.	Component	Description
1	Macbook Pro 2017	Primary operating system used by the researcher
2	Google Colaboratory [13]	Cloud-based platform for building and training the model
3	Visual Studio Code [21]	Development environment for writing program code
4	Roboflow [13]	Platform for dataset management and preprocessing

Roboflow is a specialized platform designed to assist researchers in managing datasets and preparing them for model training [13]. The dataset management capabilities of this platform streamline several essential processes, including image annotation, preprocessing, and augmentation.

d. Model Training Process

The model training was conducted using the Ultralytics library, which provides an efficient and user-friendly implementation of YOLOv8n [14], [16], [20], [24]. The model was trained with an input resolution of 640×640 pixels, a batch size of 16, and for 100 epochs. During training, monitoring was performed on loss values (box loss, cls loss, dfl loss) as well as detection accuracy. During the training stage, several data augmentation techniques were automatically applied by the Ultralytics YOLOv8 framework to increase dataset variability and improve the model's generalization performance. The primary augmentations included Horizontal Flip, which generates mirrored versions of the owl images to enhance directional robustness; Random Affine Transformations, consisting of rotation, scaling, translation, and shear adjustments that simulate geometric variations commonly found in real-world environments; Mosaic Augmentation, which combines four different images into a single training sample to improve the model's ability to detect objects in cluttered scenes and under partial visibility; and HSV (Hue, Saturation, Value) Augmentation, which modifies color intensity to replicate diverse lighting conditions such as low-light, dim, and uneven illumination inside swiftlet farming buildings [20].

Figure 2 illustrates the complete workflow of the YOLOv8n model training process, spanning from dataset preparation to the evaluation phase [25], [26]. The procedure commences with dataset collection, followed by pre-processing and data generation to ensure data readiness. Subsequently, the data splitting process is conducted to divide the data into training and testing sets. In accordance with the workflow presented in Figure 2, the dataset underwent a preprocessing stage before training. This process included resizing all images to 640×640 pixels, normalizing pixel values, and converting the annotations into the YOLO format to ensure compatibility with the Ultralytics framework. The data generation stage refers to the automatic creation of augmented training samples during runtime, where multiple variations of the original images are produced through operations such as flipping, scaling, color adjustments, and mosaic blending. These processes collectively increase dataset diversity and help the model learn robust representations of owl features in challenging environments.

 DOI: https://doi.org/10.52435/jaiit.v7i2.733

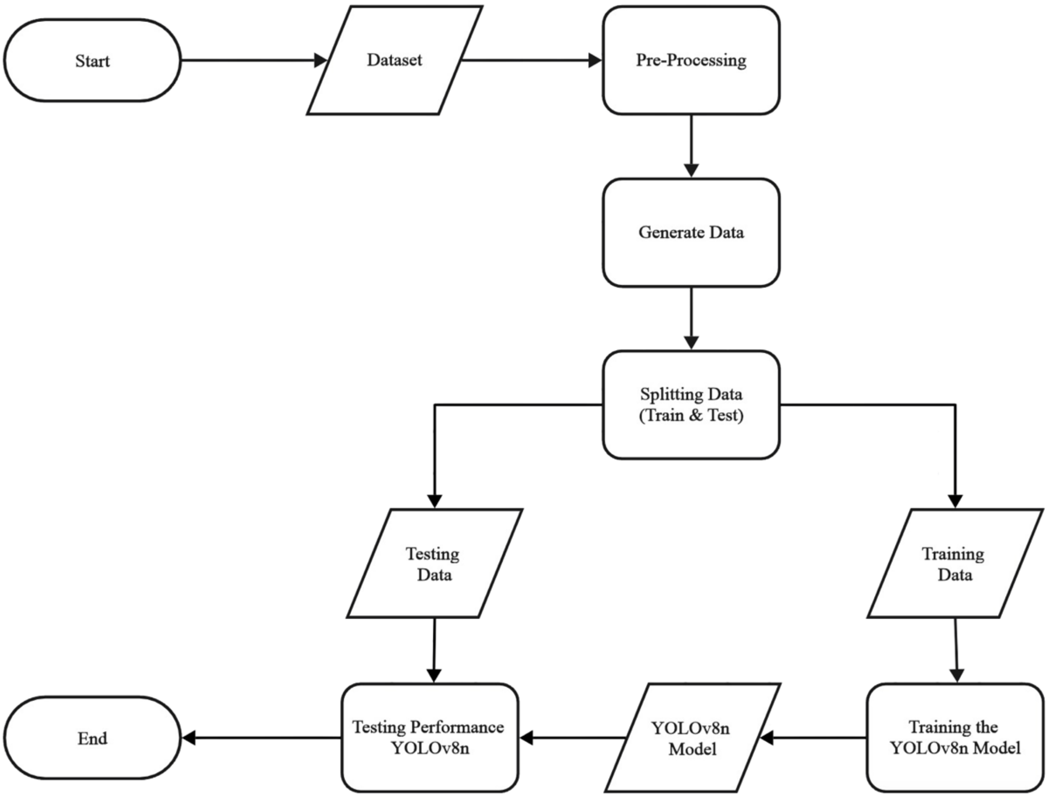

Figure 2. Model Training Pipeline [16], [20], [27]

The training data is utilized for YOLOv8n model training, while the testing data is employed for model performance evaluation [16]. The workflow presented in Figure 2 encapsulates a comprehensive and methodologically rigorous pipeline that ensures each stage of the experimental process is executed in a systematic and logically coherent manner. By clearly delineating the sequence from dataset acquisition through preprocessing, augmentation, and data partitioning, up to model training and performance evaluation, the workflow provides a transparent foundation for reproducibility and methodological integrity.

e. Model Performance Evaluation

To support quantitative evaluation, the following key metric calculation formulas were employed:

i. Precision (P)

$$P = \frac{TP}{TP + FP} \tag{1}$$

Precision measures the accuracy of positive predictions made by the model, where TP represents the number of true positive predictions, and FP denotes the number of false positive predictions[28].

ii. Recall (R)

$$R = \frac{TP}{TP + FN} \tag{2}$$

Recall describes the extent to which the model successfully identifies all relevant objects in the data, with FN representing the number of relevant objects that were not detected[29].

iii. Mean Average Precision (*mAP*)

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{3}$$

mAP (mean Average Precision) represents the average of the Average Precision (*AP*) for each class, reflecting the balance between precision and recall across various *IoU* thresholds [29]. Note that TP is True Positive, FP is False Positive, FN is False Negative, N is the number of classes, and AP is the Average Precision.

The evaluation was conducted using mAP@0.5, mAP@0.5:0.95, precision, and recall metrics on the validation data. The mAP@0.5 value indicates the model's accuracy in detecting objects at an IoU threshold ≥ 0.5, while mAP@0.5:0.95 measures the average performance across various IoU thresholds. Precision measures the accuracy of predictions, and recall measures the rate of successful detection of relevant objects.

3. Results and Discussions

The training results of the YOLOv8n model demonstrate highly promising performance in automatically detecting owl objects. One primary reason for selecting YOLOv8n is its efficiency compared to other YOLO variants [27]. YOLOv8n is the smallest and fastest version that still maintains high accuracy, making it highly suitable for real-time monitoring systems based on limited devices[20]. Another advantage of YOLOv8n lies in its use of the C2f architecture, which enables better computational efficiency compared to its predecessors. As a single-stage detector, YOLOv8n can perform detection and classification in a single, concise, and rapid process [30].

Figure 3 shows the model's inference results when processing five owl images. Each displayed blue box represents a successfully detected object prediction by the YOLOv8n model along with its class label. The label "1" indicates that the model detects the object as an owl (target class), while the label "0" indicates no detection or another class.

Based on the visualization results, it is evident that the model can consistently identify owl objects across various images, including those with dim backgrounds, low contrast, or complex lighting conditions. Although there are some predictions that are potentially incorrect (false positives/false negatives), the majority of predictions show good precision and bounding box positioning.

The dataset inherently contains low-light and visually challenging images, as the majority of owl intrusions in swiftlet farming buildings occur at night with limited illumination. Consequently, the dataset reflects realistic environmental conditions, including dark backgrounds, low contrast, and partially visible owl features. Although these images may appear visually unclear, they accurately represent the real-world scenarios in which an automated detection model must operate.

DOI: https://doi.org/10.52435/jaiit.v7i2.733

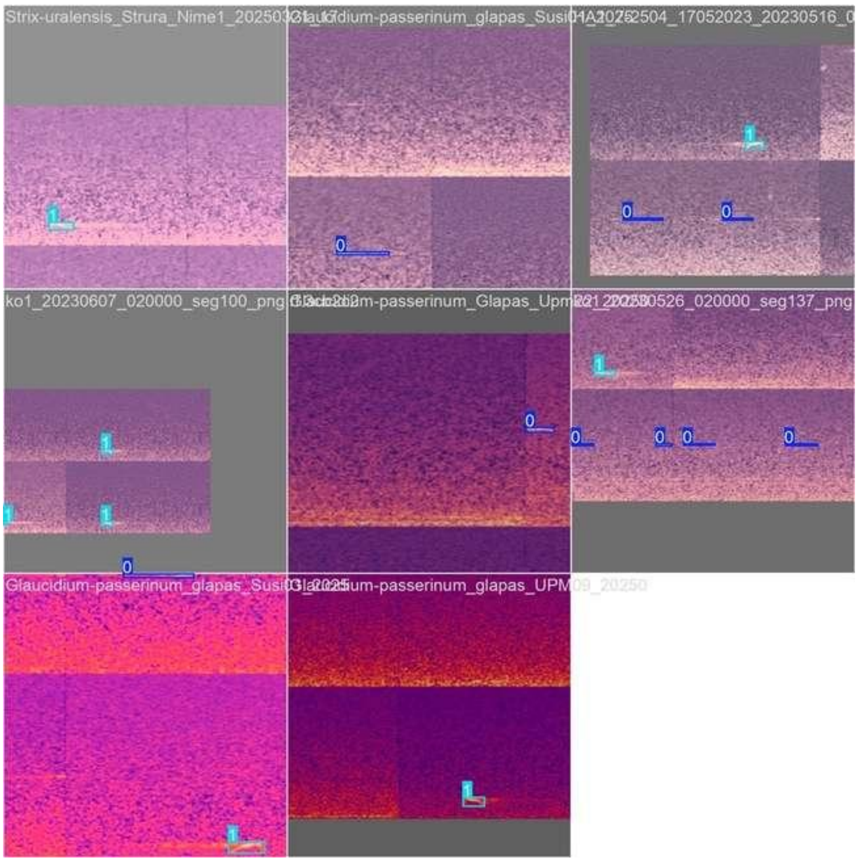

Figure 3. Model Inference Results

In Figure 4, the model was trained for 100 epochs, with each iteration producing a significant decrease in box loss, class loss, and DFL loss values. This decline reflects the model's ability to learn data patterns deeply while indicating a stable training process without overfitting. The selection of 100 epochs was based on considerations of balancing accuracy and training time efficiency. From observations of the training graph, the model's accuracy tended to stabilize after the 80th epoch; thus, 100 epochs were deemed optimal to achieve convergence without wasting computational resources.

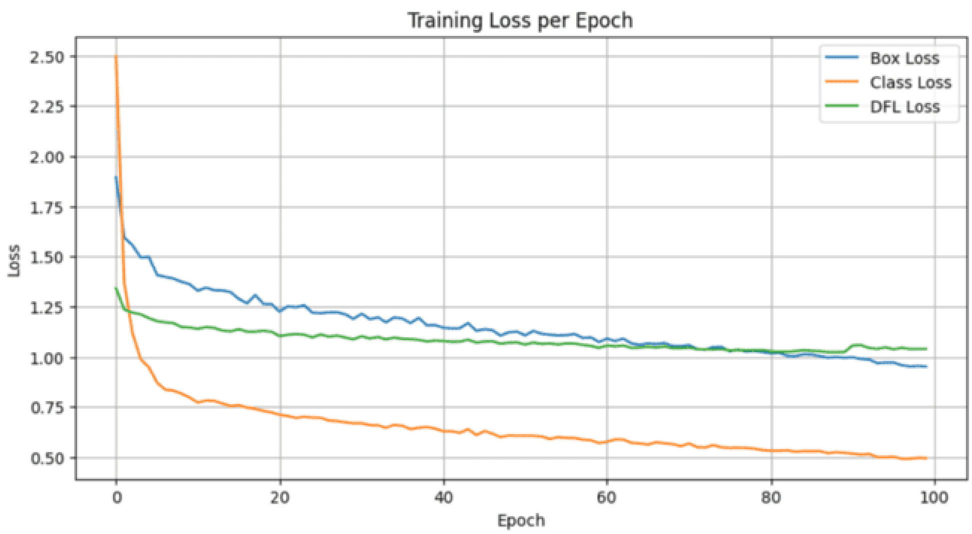

Figure 4. Training Loss per Epoch

Evaluation of the model using standard metrics demonstrates satisfactory results (Figure 5). The mAP@0.5 value of 96.82% indicates the model's capability to detect objects with very high accuracy when the Intersection over Union (IoU) threshold is set at 0.5. Meanwhile, the mAP@0.5:0.95 value of 70.5% reflects the model's average performance across various prediction difficulty levels (using IoU thresholds ranging from 0.5 to 0.95). In general, an mAP@0.5:0.95 value above 60% is considered good for single-class object detection tasks, particularly when considering the complexity of image data and diverse background conditions.

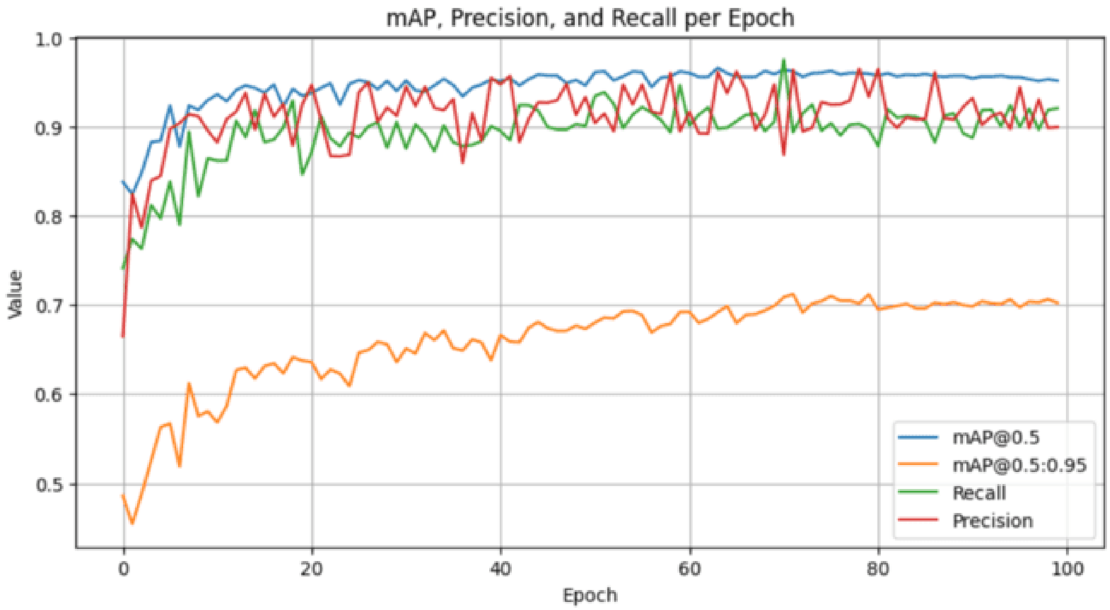

Figure 5. Modeling Results Graph

The precision and recall graphs demonstrate stability above 90% after the 20th epoch, indicating the model's excellent generalization capability on new data. This suggests that the model can not only recognize owl objects in the training data but also identify them in previously unseen conditions, as shown in Table 2. With a precision of 97.2% and a recall of 93.38%, the model exhibits very low classification error rates.

Table 2. YOLOv8n Performance Evaluation Results

No.	Metric	Value
1.	Number of Images	799 Images
2.	Number of Objects	1239 Objects
3.	Precision	97.2%
4.	Recall	93.38%
5.	mAP@0.5	96.82%
6.	mAP@0.5:0.95	70.5%

Although the YOLOv8n model achieved high precision and recall values, several failure cases were observed during qualitative evaluation. False positives generally occurred in images where background elements such as wooden beams, shadows, or structural components inside the swiftlet building resembled the silhouette of an owl. In these cases, the model incorrectly produced bounding boxes around non-owl objects, indicating that certain background textures share visual similarity with the target class. Conversely, false negatives were found in low-light images or scenarios where the owl

 DOI: https://doi.org/10.52435/jaiit.v7i2.733

appeared at a long distance or was partially occluded. Under such conditions, the owl's contours and color tones blended with the surroundings, making it difficult for the model to generate confident detections. These findings highlight that despite its strong overall performance, the model still exhibits sensitivity to illumination challenges, background complexity, and small-object detection scenarios.

To contextualize the use of YOLOv8n, a comparison with other lightweight object detection models is essential. Larger variants such as YOLOv8s generally achieve higher accuracy due to deeper and wider network architectures, but they require substantially more computational resources, making them less suitable for real-time deployment on resource-constrained devices commonly used in swiftlet farm environments. Meanwhile, earlier models such as YOLOv5n offer comparable inference speed but typically produce lower accuracy, particularly when detecting small objects under challenging lighting conditions. Given the need for a compact yet accurate detector, YOLOv8n represents an optimal trade-off by delivering strong performance while maintaining low computational requirements, supporting efficient real-time monitoring applications in practical field settings.

The noticeable decrease from mAP@0.5 (96.82%) to mAP@0.5:0.95 (70.5%) reflects the model's reduced bounding box precision at higher IoU thresholds. This behavior is expected for lightweight architectures such as YOLOv8n, which typically struggle to maintain highly accurate box localization under stringent evaluation criteria. The owl images used in this study often exhibit low illumination, cluttered backgrounds, and varied shooting angles, contributing to less precise bounding box alignment. As the IoU requirement increases from 0.5 to 0.95, even small deviations in box placement significantly lower the overall score. Therefore, the observed performance drop highlights the inherent trade-off between model compactness and localization accuracy while still demonstrating that YOLOv8n performs reliably within the practical accuracy requirements of real-time surveillance applications.

These results demonstrate that the implementation of YOLOv8n is highly relevant for owl detection in swiftlet farming buildings. The combination of high accuracy, inference efficiency, and compact model size provides flexibility for integration with camera systems or microprocessors in the field. This system can be developed into an automated early-warning solution that can reduce the risk of predator attacks in real-time and support the sustainability of swiftlet farming ecosystems optimally.

4. Conclusion

This study successfully implemented the YOLOv8n model for automatic owl detection in swiftlet farming environments. The model was trained using an annotated owl image dataset in YOLO format within a cloud-based Google Colab training environment. Evaluation results demonstrate that the model achieved a precision of 97.2%, a recall of 93.38%, mAP@0.5 of 96.82%, and mAP@0.5:0.95 of 70.5%. These values indicate the model's capability to detect owl objects with high accuracy, even under complex background conditions and varying lighting. YOLOv8n proved capable of combining computational efficiency with reliable detection accuracy. Its lightweight nature enables deployment on resource-constrained devices such as edge devices or real-time camera-based surveillance systems. Additionally, the fast training and inference processes make this model relevant for monitoring applications requiring rapid detection and high precision.

Based on the obtained results, this model has significant potential for integration into early warning systems in swiftlet farms. The developed system is expected to assist farmers in identifying owl presence earlier, enabling prompt preventive measures. Future research can enhance this system through integration with hardware such as smart cameras or microcontrollers, as well as expanding detection coverage to multiple predator types to comprehensively improve farm security system resilience.

References

[1] S. Kurnia *et al.*, "JEPIN (Jurnal Edukasi dan Penelitian Informatika) Perancangan Prototipe Sistem Kontrol Suhu dan Kelembaban pada Gedung Walet dengan Mikrokontroler Berbasis Mobile".

[2] C. Hakim *et al.*, "Studi Budidaya Burung Walet di Wilayah Gorontalo Kawasan Teluk Tominis," Pontianak, Jan. 2024. [Online]. Available: https://journal.ikippgriptk.ac.id/index.php/snpp/article/view/6984

[3] Yovi Apridiansyah, A. Wijaya, Pahrizal, Rozali Toyib, and Arif Setiawan, "Pengolahan Citra Berbasis Video Proccesing dengan Metode Frame Difference untuk Deteksi Gerak," *Journal of Applied Computer Science and Technology*, vol. 5, no. 1, pp. 81–89, Jun. 2024, doi: 10.52158/jacost.v5i1.790.

[4] Y. Akkem, S. K. Biswas, and A. Varanasi, "Smart farming using artificial intelligence: A review," *Eng Appl Artif Intell*, vol. 120, Apr. 2023, doi: 10.1016/j.engappai.2023.105899.

[5] M. Mahmud, M. S. Kaiser, A. Hussain, and S. Vassanelli, "Applications of Deep Learning and Reinforcement Learning to Biological Data," *IEEE Trans Neural Netw Learn Syst*, vol. 29, no. 6, pp. 2063–2079, Jun. 2018, doi: 10.1109/TNNLS.2018.2790388.

[6] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: Exceeding YOLO Series in 2021 V100 batch 1 Latency (ms) YOLOX-L YOLOv5-L YOLOX-DarkNet53 YOLOv5-Darknet53 EfficientDet5 COCO AP (%) Number of parameters (M) Figure 1: Speed-accuracy trade-off of accurate models (top) and Size-accuracy curve of lite models on mobile devices (bottom) for YOLOX and other state-of-the-art object detectors." [Online]. Available: https://github.com/ultralytics/yolov3

[7] S. Xu *et al.*, "PP-YOLOE: An evolved version of YOLO," Dec. 2022, [Online]. Available: http://arxiv.org/abs/2203.16250

[8] M. Hussain, "YOLOv1 to v8: Unveiling Each Variant-A Comprehensive Review of YOLO," *IEEE Access*, vol. 12, pp. 42816–42833, 2024, doi: 10.1109/ACCESS.2024.3378568.

[9] W. Fang, L. Wang, and P. Ren, "Tinier-YOLO: A Real-Time Object Detection Method for Constrained Environments," *IEEE Access*, vol. 8, pp. 1935–1944, 2020, doi: 10.1109/ACCESS.2019.2961959.

[10] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," in *Procedia Computer Science*, Elsevier B.V., 2021, pp. 1066–1073. doi: 10.1016/j.procs.2022.01.135.

[11] X. Chen, C. Liu, S. Wang, and X. Deng, "LSI-YOLOv8: An Improved Rapid and High Accuracy Landslide Identification Model Based on YOLOv8 From Remote Sensing Images," *IEEE Access*, vol. 12, pp. 97739–97751, 2024, doi: 10.1109/ACCESS.2024.3426040.

[12] H. Liu, R. Hu, H. Dong, and Z. Liu, "SFC-YOLOv8: Enhanced Strip Steel Surface Defect Detection Using Spatial-Frequency Domain-Optimized YOLOv8," *IEEE Trans Instrum Meas*, vol. 74, 2025, doi: 10.1109/TIM.2025.3548193.

[13] V. V. Kukartsev, R. A. Ageev, A. S. Borodulin, A. P. Gantimurov, and I. I. Kleshko, "Deep Learning for Object Detection in Images Development and Evaluation of the YOLOv8 Model Using Ultralytics and Roboflow Libraries," in *Lecture Notes in Networks and Systems*, Springer Science and Business Media Deutschland GmbH, 2024, pp. 629–637. doi: 10.1007/978-3-031-70285-3_48.

[14] G. A. Pereira, "Fall Detection for Industrial Setups Using YOLOv8 Variants," Aug. 2024, [Online]. Available: http://arxiv.org/abs/2408.04605

[15] P. K. Pativada, "Real-time detection and classification of plant seeds using YOLOv8 object detection model," 2024.

[16] Z. Zhang *et al.*, "An Improved YOLOv8n Used for Fish Detection in Natural Water Environments," *Animals*, vol. 14, no. 14, Jul. 2024, doi: 10.3390/ani14142022.

[17] S. Guan and S. Lv, "MCG-YOLOv8n: An Enhanced YOLOv8n Model for Segmentation of Adherent Buckwheat Seeds," *IEEE Access*, vol. 12, pp. 131968–131981, 2024, doi: 10.1109/ACCESS.2024.3460174.

[18] Y. Liu, Q. Ye, L. Sun, and Z. Wu, "SOD-YOLOv8n: Small Object Detection in Remote Sensing Images Based on YOLOv8n," *IEEE Geoscience and Remote Sensing Letters*, vol. 22, 2025, doi: 10.1109/LGRS.2025.3567362.

[19] Q. Liu *et al.*, "DSW-YOLOv8n: A New Underwater Target Detection Algorithm Based on Improved YOLOv8n," *Electronics (Switzerland)*, vol. 12, no. 18, Sep. 2023, doi: 10.3390/electronics12183892.

[20] L. Chen, G. Li, S. Zhang, W. Mao, and M. Zhang, "YOLO-SAG: An improved wildlife object detection algorithm based on YOLOv8n," *Ecol Inform*, vol. 83, Nov. 2024, doi: 10.1016/j.ecoinf.2024.102791.

[21] P. Masci and C. A. Muñoz, "An integrated development environment for the prototype verification system," in *Electronic Proceedings in Theoretical Computer Science, EPTCS*, Open Publishing Association, Dec. 2019, pp. 35–49. doi: 10.4204/EPTCS.310.5.

[22] T. S. Gunawan *et al.*, "Development of video-based emotion recognition using deep learning with Google Colab," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 5, pp. 2463–2471, Oct. 2020, doi: 10.12928/TELKOMNIKA.v18i5.16717.

[23] K. R. M. Leino and V. Wüstholz, "The Dafny integrated development environment," in *Electronic Proceedings in Theoretical Computer Science, EPTCS*, Open Publishing Association, Apr. 2014, pp. 3–15. doi: 10.4204/EPTCS.149.2.

[24] P. K. Pativada, "Real-time detection and classification of plant seeds using YOLOv8 object detection model," 2024.

[25] Q. Liu *et al.*, "DSW-YOLOv8n: A New Underwater Target Detection Algorithm Based on Improved YOLOv8n," *Electronics (Switzerland)*, vol. 12, no. 18, Sep. 2023, doi: 10.3390/electronics12183892.

[26] Y. Fan, L. Zhang, and P. Li, "A Lightweight Model of Underwater Object Detection Based on YOLOv8n for an Edge Computing Platform," *J Mar Sci Eng*, vol. 12, no. 5, May 2024, doi: 10.3390/jmse12050697.

[27] B. Zhao, Q. Zhang, Y. Liu, Y. Cui, and B. Zhou, "Detection Method for Rice Seedling Planting Conditions Based on Image Processing and an Improved YOLOv8n Model," *Applied Sciences (Switzerland)*, vol. 14, no. 6, Mar. 2024, doi: 10.3390/app14062575.

[28] M. Zhu, "Recall, Precision and Average Precision," 2004.

[29] L. Torgo and R. Ribeiro, “Precision and recall for regression,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2009, pp. 332–346. doi: 10.1007/978-3-642-04747-3_26.

[30] N. Ma, Y. Wu, Y. Bo, and H. Yan, “Chili Pepper Object Detection Method Based on Improved YOLOv8n,” *Plants*, vol. 13, no. 17, Sep. 2024, doi: 10.3390/plants13172402.