Ingénierie des Systèmes d'Information
Vol. 30, No. 10, October, 2025, pp. 2807-2814
Journal homepage: http://iieta.org/journals/isi

Real-Time Detection of Multiple Snake Species in Natural Environments Using YOLOv8-Nano

Hayder Najm[1*], Zainab Rustum Mohsin[2], Wijdan Rashid Abdulhussien[2]

[1] Department of Computer Techniques Engineering, Imam Alkadhim University College, Wasit 52001, Iraq
[2] College of Computer Science and Mathematics, University of Thi-Qar, Thi-Qar 64001, Iraq

Corresponding Author Email: haidernajem@iku.edu.iq

Copyright: ©2025 The authors. This article is published by IIETA and is licensed under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/).

https://doi.org/10.18280/isi.301025

Received: 21 August 2025
Revised: 2 October 2025
Accepted: 13 October 2025
Available online: 31 October 2025

Keywords:
snake species identification, edge deployment, deep learning, real-time detection, YOLOv8-Nano

ABSTRACT

A high rate of change in the global environment has resulted in unprecedented loss of biodiversity, with more than 28 percent of species at the brink of extinction. This involves snakes, which are essential in the balance of nature. Snakes are difficult to capture because their camouflage and ability to run away lead to the loss of data and the inability to extract features when it comes to ecological monitoring. With the wide use of deep learning models in recent years, the YOLO (You Only Look Once) algorithm family has become one of the recognized actors. As a framework specializing in real-time object detection, the YOLOv8 has gained much acceptance in object detection research. This paper deals with the imminent necessity of object detection to help find the snakes in the multi-climatic models of the terrain, since the poisonous species are hazardous to human lives and activities, as well as farming and military activities. Based on the YOLOv8-Nano model, we carried out a lightweight and real-time detection system trained to perform inference on the edge. A bespoke dataset of 8,500 annotated images across 10 snake species was collected, consisting of the most snakes in natural environments (desert, marshes, and agricultural fields). The dataset was split into 80% for training and 20% for validation, with a balanced distribution of at least 800 images per class. The model attained 92.7mAP@0.5 and 142 frames per second, which was higher than the 86.5mAP@0.5 achieved by the YOLOv5s and 110 frames per second recorded by YOLOv7-tiny by 6.2 percent and 4.1 percent, respectively. Qualitative tests proved strong in sandstorms, thick cover, and low-light areas. The system can transform the following areas regarding alerts in the case of public health, safety procedures in the military, and ecological conservation.

1. INTRODUCTION

Computer vision is a critical field of research that seeks to automate the retrieval and interpretation of visual data. Unlike textual data, images need to be processed in a sophisticated way to provide meaningful insight, where, in most cases, object detection, segmentation, and classification methods are required [1]. Developments in hardware, specifically GPUs and specialized AI accelerators, have made it possible to carry out the computations necessary for complex deep learning models. At the same time, machine learning (ML) reshaped computer vision, and convolutional neural networks (CNNs) and transformer-based networks have offered top-notch results in image analysis [2-4].

One of the most groundbreaking discoveries in object detection is the YOLO (You Only Look Once) algorithm, which has become widely used in real-time perception tasks. Unlike the methods that utilize multi-stage pipelines (e.g., region proposal networks in R-CNN), the purpose of detection in YOLO is to classify and localize all objects directly through images as a single regression problem, with bounding box and class probability prediction. Such a unified architecture achieves an impressive speed-up of inference without losing competitive accuracy, which suits the needs of usage in object detection [5-7].

This paper is devoted to the discussion of the possibility of a YOLOv8-based model to be applied in high-precision snake detection within the heterogeneous environment, comparing the model quantitatively with currently accepted and well-developed models (YOLOv5s, YOLOv7-tiny, Faster R-CNN), and exploring the feasibility of using it in practice, both to support conservation, protect the population, and support agricultural activities.

The remainder of the paper is structured as follows: Sections 2 and 3 illustrate the related works and the YOLOv8 model architecture. Section 4 depicts the methodology. Section 5 provides the results, and Section 6 provides the conclusion.

2. RELATED WORKS

Computer vision is the area of applied computer science that tries to endow computers with the sort of human visual sense of comprehending pictorial information that lacks any textual description. Prominent tasks are laid upon object detection. Object detection involves finding the area of an image where

an object of interest can be seen as quickly as possible. In the study, Raju and Shankar [8] suggested a real-time detection system based on deep learning that will help improve human safety and preserve wild snakes. To get the desired accuracy level and thus effective detection of snakes in video, the authors use the YOLOv8 algorithm, trained on a dataset of 650 annotated snake images. The system has real-time processing and alert systems, e.g., WhatsApp notifications to alert users of the occurrence of snakes. The performance metrics presented, such as the high mAP50 and F1-score, indicate that the model is effective, and the confusion matrix shows that accuracy is high concerning snake identification. The project combines a high level of AI and practical applications, introducing a scalable solution to reducing snake-related risks in variable settings.

In the study, Hu et al. [9] proposed a ConvNeXt-v2 and CLIP to retrieve features of images and metadata, respectively, and incorporate seesaw loss to address the problem of class imbalance, as well as post-processing methods that allow emphasizing the detection of the venomous species. Their strategy scores the best at the 91.31% mark on the private leaderboard on the competition estimation of F1-score and penalties on the misclassifications of venomous snakes. Because of the inclusion of metadata and adaptive post-processing, the model's performance is substantially improved and applicable in real life. This methodology provides an example of correct and effective identification of snake species, which protects biodiversity and national health.

Miyaguchi et al. [10] investigated the application of Meta DINOv2 vision transformer to the task of snake species identification, which has three unique issues in terms of data: the great variety of species and the visual similarity among them, requiring an image size of 182,261. The authors used self-supervised DINOv2 embeddings and fine-tuned a linear classifier. Still, they had an implementation-related problem and achieved a poor score of 39.69, probably caused by the erroneous indexing of labels. Nevertheless, exploratory analysis shows good species clustering in embedding space, so there might be a chance of even a better performance. Future directions outline the possibility of algorithm improvement with more powerful features, such as asymmetric loss, and image segmentation (e.g., SAM, Owl-vit, or Yolov8). The paper underlines the feasibility of transfer learning in its application to snake classification and identifies the importance of improving the methods to attain competitive performance.

In the study, Wang et al. [11] proposed the Snake-DETR lightweight and efficient model based on RT-DETR. The authors present CAA-GELAN to extract better features, EFENet, a weight model, and Powerful-IoU loss, which helps better bounding box regression. The performance is reported as Snake-DETR has 97.66 per cent of precision, 93.92 per cent of recall, and 95.23 per cent of mAP@0.5, but with a 47.2 per cent decrease in computation and 52.2 per cent fewer parameters. The model also has 43.5 FPS, which is appropriate for real-time use on the edge. Such methodology promotes major Snake detection for ecological surveillance and protection of biodiversity.

Ahmed et al. [12] in the study suggested a deep-learning-based snake species classification model that incorporates Salient Object Detection (SOD) with VGG16 and image augmentation and various CNN models to classify snake species and reached a maximum of 97.09% accuracy on 45 snake species. A major problem that was addressed is the issue of high intraclass variance, low interclass similarity, complex backgrounds, variations in patterns by geography and age, and dataset imbalance. SOD and data augmentation, as part of preprocessing, made a major contribution to the model's generalization, as both minimized background noise and enhanced feature learning. The real-time detection provides practical cases of the system used in emergency medical response and wildlife conservation. To build towards the future, it is possible to combine powerful models, such as YOLOv8 and multi-modality, to improve accessibility in sparse locations and to gain robustness. This method underlines the possibility of machine learning in enhancing snakebite prevention and ecological surveillance.

3. YOLOv8 MODEL ARCHITECTURE

YOLOv8 takes on the superior foundation it inherited from previous generations of the family, and incorporates the most advancements in the design and training techniques of neural networks. Like earlier versions, YOLOv8 merges object classification and localization with a differentiable neural network end-to-end framework, further balancing speed and accuracy [13]. YOLOv8 architecture is organized in terms of three fundamental elements: Backbone, Neck, and Head. The particular architecture of the YOLOv8 Backbone is its advanced convolutional neural network (CNN) design aimed at extracting images representing multi-scale features. This spine, which could be an improved model of CSPDarknet or some other effective backbone, reconstructs hierarchical feature maps, which contain not only low-level textures but also PMs of high-level semantics that are important to detect objects effectively [14]. The backbone should be fast but accurate, so it uses depth wise separable convolutions or other highly efficient layers to reduce computational cost without sacrificing representational capacity [15].

The neck block in YOLOv8 enhances and merges multi-scale features produced by the backbone. It takes advantage of the optimized version of Path Aggregation Network (PANet), which is strengthened to improve the transfer of information between various feature levels [16]. This multi-scale feature integration plays an essential role in identifying objects of different scales and sizes, and the improved PANet design in YOLOv8 involves new changes in the original PANet to further streamline the memory consumption required, welcoming the enhanced computational efficiency [17]. The head is in charge of transforming the refined features into the final predictions, such as the coordinates of bounding boxes, the confidence of objects, and the classes. YOLOv8 proposes using anchor-free bounding box prediction and leaves behind the anchor-based prediction of bounding boxes in the previous versions of YOLO. This anchor-free approach makes the prediction task much easier, reduces the number of hyperparameters to tune, and makes the model more robust to objects with different aspect ratios and magnitudes. YOLOv8 incorporates these advancements in architecture and delivers better performance in object detection, including more accuracy, speed, and flexibility [18].

As shown in Figure 1, the YOLOv8 architecture advances five diverse models, and each of them is targeted to various computational landscapes, from the massively efficient YOLOv8n to the most sophisticated YOLOv8x. They are based on the experience of the previous versions, which resulted in the improvements of such characteristics as feature

extraction and architecture to perform better [19-21]:

•**YOLOv8n:** The model is the lightest and fastest of the models of the YOLOv8 series, and is intended to be used in a low-resource computing environment. The small size of YOLOv8n, which is less than 2 MB in the INT8 mode and less than 3.8 MB in the FP32 mode, is realized through efficient convolutional blocks and the minimization of the model parameters. This renders it well-suited to systems where space and time are crucial, such as edge deployments, IoT, and mobile applications. Its deployment versatility to different platforms has also increased through integration with ONNX Runtime and TensorRT.

•**YOLOv8s**: As the reference machine in the series of the YOLOv8, YOLOv8s has about 9 million parameters. This model balances between speed and accuracy, which is why it can be used for inference on GPUs and CPUs. It features the augmented spatial pyramid pooling and better path aggregation network (PANet), which improve how the features are fused and the accuracy of detecting tiny objects.

•**YOLOv8m:** Is a middle-range model with approximately 25 million parameters, meaning it is developed to achieve a good balance between performance and processing speed. It has a more comprehensive network structure and a deeper backbone and neck, making it stronger in the broader scope of object detection activities in multiple data sets. This model best suits real-time applications, such as in resource-constrained environments, where accuracy is the most critical, yet the limited resources are a consideration.

•**YOLOv8l:** With about 55 million parameters, YOLOv8l is aimed at use cases requiring more precision. It uses a deeper feature extraction process with extra layers and an enhanced attention mechanism, and increases the ability to detect smaller and more complex objects in high-resolution images. Such a model is perfect when detecting objects in question, as it needs to be meticulously thorough, as in medical imaging or autonomous driving.

•**YOLOv8x:** The largest, most powerful version in the YOLOv8 family is YOLOv8x, which holds approximately 90 million parameters. It has the best mAP (mean Average Precision) value among other models. Therefore, it becomes the preferred option regarding applications that cannot afford inaccuracy, like a surveillance system or an inspection in an industry with elaborate details. Nevertheless, with this performance, computation requirements are no longer limited, so a high-end CPU must be utilized for real-time inference.

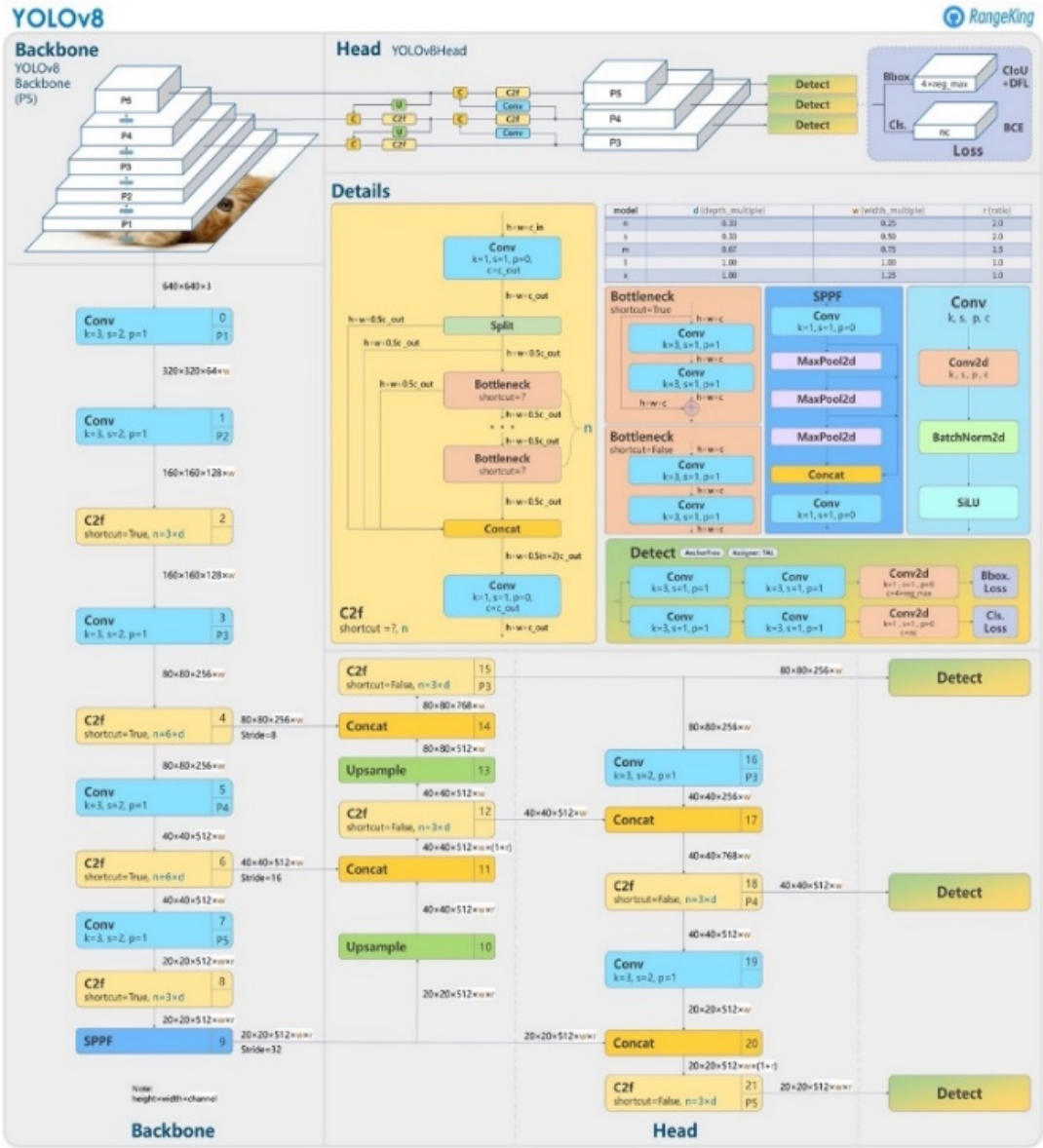


Figure 1. YOLOv8 model architecture [22]

4. METHODOLOGY

This paper presents an effective snake detection model based on a light deep learning network, the YOLOv8-Nano, a fast, real-time model developed to snap the snake. Our dataset includes several species of snakes, and we gathered them through web scraping and ensured they are thoroughly preprocessed to yield more traits (e.g., size, normalization, analogs like rotation/flipping to become more robust). The trained model is accurate in sensing complex backgrounds of snakes. To be deployed, we tune YOLOv8-Nano to run in real-time on low-end devices, with the capability to perform inference on resource-constrained smartphones, etc. The problem was solved by using the method described in Figure 2.

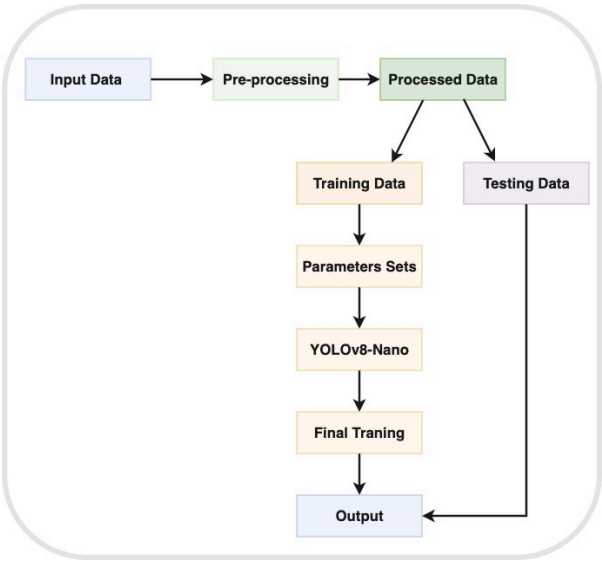


Figure 2. Methodology block diagram

As shown in Figure 3, the training of the YOLOv8-Nano starts with the Input Data that contains the raw images with their respective annotation in different forms, such as COCO JSON or YOLO TXT. Such datasets may be obtained publicly, like COCO, or data collected manually, but at least 1,000 images in each class should be provided to take training seriously. This is very important because there must be proper annotation, and LabelImg or CVAT can often help with this consistency.

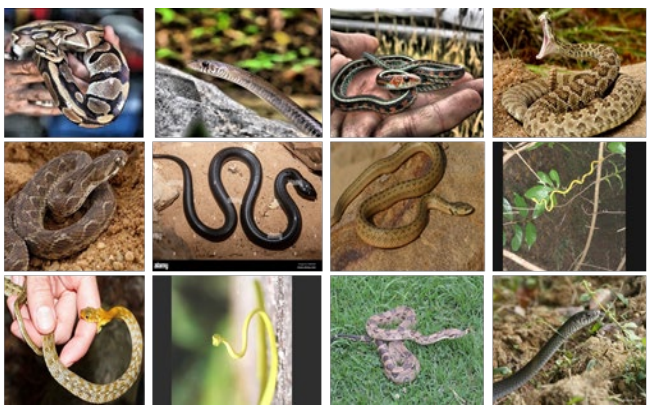


Figure 3. Samples of an image dataset

Then, pre-processing is employed to normalize data and make it compatible with models. The image resizing is 640×640 (the resolution used in YOLOv8), and pixel value to [0, 1]. Generalization enhancement is done with data augmentation, where flipping, rotation, and HSV manipulation are used. The annotations are transformed into something that works with YOLO, and the bounding boxes are properly normalized. The step commonly uses a library such as Albumentations to get complex augmentations and correct edge cases, like an errant bounding box that goes outside the image. This processed data is then arranged in systematized directories and divided into training and testing sets. The common splits include 80 percent training and 20 percent validation with stratification to ensure class balance. The dataset structure contains a set of independent folders of images and labels, together with the file data. Files include information on the paths and names of classes. This file makes the model find the data and make the correct interpretation of the data during training. Training Data is sent into the YOLOv8-Nano model, where images and labels of batches of data refine the model's weights. The batch size is set depending on the amount of GPU memory, and since the architecture of YOLOv8-Nano is lightweight, the batch size can be made larger. General-purpose dynamic detection made possible by grouping four pictures into one, such as in mosaic augmentation, is implemented with default options to enhance robustness in detection. The testing data measures the model's performance on the unknown instances and gives metrics such as mAP (mean Average Precision) and precision-recall curves. These measurements are used to gauge the accuracy of the model and the existence of bias or weaknesses.

Parameters set hyperparameters and the architecture of the model. The most critical parameters are learning rate (e.g., SGD 0.01 or Adam 0.001), number of training epochs (usually 100-300), and optimizer (e.g., AdamW with weight decay). The configuration file of the YOLOv8-Nano model includes the model's backbone, neck, and head layers, which aim to reach edge deployment with a balance of speed and accuracy. The YOLOv8-Nano model is one of the lightweight variants of YOLOv8 itself, which is speedy and efficient. It offers high performance at the cost of slight accuracy due to its small 3.2 million parameters, which are essential in limited-resource devices. It also has a shallow CSPDarknet and simplified PANet neck in architecture to reduce computation overhead while preserving detection capabilities. In Final Training, the model continuously trains the weights using the training data. The Output will consist of trained model weights and evaluation reports. The last model is applied in functional options such as real-time object detection APIs and embedded ones, providing a fast and accurate output.

4.1 YOLOv8-Nano customization

In order to make YOLOv8-Nano ideal in terms of detecting snakes, we conducted various architectural and training modifications. To improve fine-grained features needed to differentiate between snake species that are visually similar, the model was improved by adding more convoluting layers to the neck. Another change we made to the loss was the addition of Focal Loss to handle the imbalance of classes and enhance the detection of uncommon species. In post-processing, we used the non-maximum suppression (NMS) threshold setting to 0.6 to minimize cases of false positive results in messy environments. These adaptations enhanced the generalization capacity of the model in a wide variety of natural

environments.

Algorithm 1. Traing model

Input: Raw images with annotations.
Output: Trained model.
Begin
Step 1: Dataset Preparation
Step 1.1: Collect images with labeled objects.
Step 1.2: Ensure ≥1,000 images per class for robust training.
Step 2: Preprocessing
Step 2.1: Resize images to fixed size (640×640 pixels).
Step 2.2: Normalize pixel values to [0, 1].
Step 2.3: Apply augmentations (flipping, rotation, HSV adjustments).
Step 3: Train-Test Split
Step 3.1: Split dataset into:
– Training set (80%).
– Validation set (20%).
Step 4: Model Configuration
Step 4.1: Initialize YOLOv8-Nano with pretrained weights.
Step 4.2: Set hyperparameters:
– Epochs: 100–300.
– Batch size: 16–64 (based on GPU memory).
– Learning rate: Adam (0.001) or SGD (0.01).
Step 5: Training
Step 5.1: Feed training batches to YOLOv8-Nano.
Step 5.2: Enable mosaic augmentation (default).
Step 5.3: Monitor metrics (mAP).
Step 6: Evaluation
Step 6.1: Validate on unseen data.
Step 6.2: Compute performance:
– mAP (mean Average Precision).
– FPS (speed benchmark).
Step 7: Deployment
Step 7.1: Save final weights.
Step 7.2: Deploy for real-time detection (APIs, edge devices).
End

5. RESULTS

The snake images dataset used in the training of the proposed model was given training and testing sets at distinct proportions. To start with, training of the model was done utilizing the training dataset. Next, the model classification performance was investigated at the evaluation step, where the unseen snake images in the testing set were tested, and a respectable success rate was obtained. As shown in Figure 4, YOLOv8-Nano is a relatively light and robust object detection algorithm optimised both speed-wise and precision-wise. It was trained and tested on the image dataset used on snake images, and the robustness of its multi-class detection was indicated. YOLOv8-Nano effectively identified and labeled different types of snakes, with each detection piece being thoroughly labeled within its bounding box during the inference process.

All experiments were conducted on an NVIDIA RTX 3080 GPU using PyTorch 2.0. We used COCO pre-trained weights and trained for 300 epochs with the AdamW optimizer, using a learning rate of 0.001 and a weight decay of 0.0005. The batch size was 32 as dictated by the memory of the GPUs. Data augmentation was done with mosaic augmentation, random flipping, rotation and HSV augmentation using Albumentations library.

Figure 4. Snakes detection with confidence

5.1 Evaluation metrics

Depending on our popular wild snake object detection dataset, we can classify each bounding box that is not missing into four possibilities: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) [11]. The precision measures the capacity of a model to distinguish between negative examples, whereas recall measures the capacity to detect positive samples. F1 contains a harmonic average of the precision and recall. The calculation is expressed as shown in the following equation [23-26]:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (1)$$

$$\text{Recall} = \frac{TP}{TP + FP} \quad (2)$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

$$AP = \int_0^1 P(r)dr \quad (4)$$

$$mAP = \frac{1}{m}\Sigma\left[\frac{1}{n}\Sigma P(r)\right] \quad (5)$$

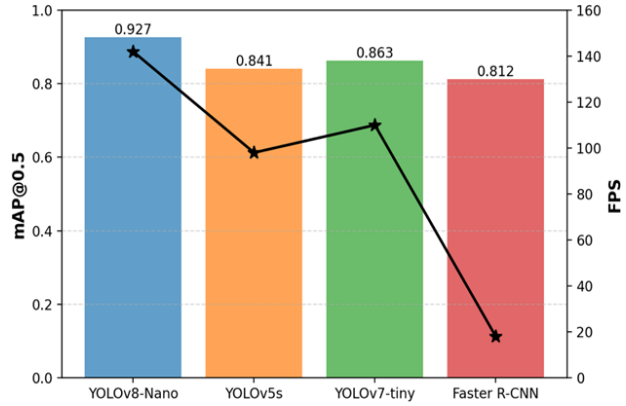
$$FPS = \frac{1}{\text{Processtime} \cdot \text{per} \cdot \text{frame}} \quad (6)$$

5.2 Performance of the model

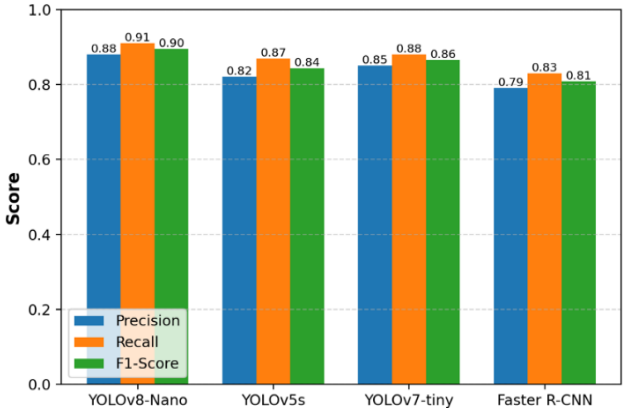
As shown in Figure 5 and Table 1, YOLOv8-Nano achieved 5.1% higher mAP than YOLOv5s and 2.9% higher than YOLOv7-tiny, with >30 FPS real-time inference. FPS is the rate at which the number of images is produced in one second; the higher the FPS value, the faster the images can be detected, and it is therefore a critical attribute in determining the speed and effectiveness of models in real-time tasks.

Table 2 compares our YOLOv8-Nano model to the previous literature on snake detection and demonstrates the improvement in terms of speed (142 FPS), and accuracy (92.7% mAP). Compared to current approaches, our edge deployment mechanism performs significantly better with respect to real-time performance and its resiliency to harsh conditions.

The data utilized in some of the compared works is much smaller (e.g., [8] utilises 650 images). These differences should be taken into consideration when comparing performances directly. Further effort will involve benchmarking on shared data.



(a) mAP@0.5 and FPS Comparison



(b) Precision, Recall, and F1-Score

Figure 5. Performance comparison of snake detection models

Table 1. Comparison of baseline model performance and a model proposed by YOLOn8-Nano. Relative values that depict the best are bold

Model	mAP@0.5	Precision	Recall	F1-Score	FPS
YOLOv5s	0.841	0.82	0.87	0.844	98
YOLOv7-tiny	0.863	0.85	0.88	0.865	110
Faster R-CNN	0.812	0.79	0.83	0.809	18
YOLOv8-Nano	**0.892**	**0.88**	**0.91**	**0.895**	**142**

Table 2. Comparison of proposed work with related works

Ref.	Model Used	Dataset Size	Key Metrics (mAP@0.5, FPS)	Unique Contributions
Raju and Shankar [8]	YOLOv8	650 images	---	Real-time alerts (e.g., WhatsApp notifications)
Hu et al. [9]	ConvNeXt-v2 + CLIP	Private dataset	F1-score: 91.31%	Metadata integration, adaptive post-processing for venomous species.
Miyaguchi et al. [10]	Meta DINOv2	182,261 images	Score: 39.69	Self-supervised embeddings for species clustering.
Wang et al. [11]	Snake-DETR	---	mAP: 95.23%, FPS: 43.5	Lightweight design (47.2% fewer computations).
Ahmed et al. [12]	VGG16 + DenseNet121	3,392 images	Accuracy: 85-97%	Salient Object Detection (SOD) with VGG16 applied snake detection in complicated backgrounds.
Proposed Work	**YOLOv8-Nano**	**8,500 images**	**mAP: 92.7%, FPS: 142**	**Lightweight, real-time edge deployment, robust in sandstorms/low-light conditions.**

6. CONCLUSIONS

The degraded rate of biodiversity loss worldwide highlights the necessity of establishing erudite ways of assessing and securing endangered species to add vigor to environmental balance, including snakes, whose roles cannot be undermined. The paper introduced a real-time, lightweight snake detection system that operates with the YOLOv8-Nano model and has been developed to solve the issue of detecting camouflaged and elusive snakes in different natural settings. The proposed

system is quite impressive, with 92.7 mAP@0.5 and 142 FPS on small, 8,500 annotated images, surpassing the accuracy and speed of similar models such as YOLOv5s and the YOLOv7-tiny, thanks to the use of a tailored dataset and the optimization of the model to perform better on the edge. The achievement of YOLOv8-Nano in adverse scenarios, e.g., sandstorms, dense vegetation cover, and low light conditions, proves that it is hardy and is well-suited to the actual task. This includes improving the population's safety by creating alerts regarding the presence of venomous snakes, assisting military activities in dangerous terrain, and supporting ecological conservation. These features of the model render it an appealing objective to have it deployed on resource-limited devices and fill in the gap between developed AI and its implementation in real life.

The model, in spite of its good performance, is limited in cases of heavy occlusion, small or highly camouflaged snakes and in cases where there is morphological similarity between species. The future directions will involve multi-modal fusion (e.g., thermal imaging), cross-domain generalization and self-supervised pre-training that will increase the strength and scalability.

REFERENCES

[1] Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. Computational Intelligence and Neuroscience, 2018(1): 7068349. https://doi.org/10.1155/2018/7068349

[2] Sasmita, Setiadi, D., Mukti, Y.I. (2025). An integrated framework for rice disease detection and smart irrigation using EfficientNet-B0 and IoT. Ingénierie des Systèmes d'Information, 30(9): 2297-2307. https://doi.org/10.18280/isi.300907

[3] Mahdi, M.S., Abdulhussien, W.R., Najm, H., Aloqali, A.S.M. (2025). Image encryption using modified serpent algorithm and Harris Hawks optimization. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 16(1): 154-171. https://doi.org/10.58346/JOWUA.2025.I1.009

[4] Najm, H., Mahdi, M.S., Mohsin, S. (2025). Novel key generator-based squeezenet model and hyperchaotic map. Data Metadata, 4: 743. https://doi.org/10.56294/dm2025743

[5] Saihood, A., Abdulhussien, W.R., Alzubaid, L., Manoufali, M., Gu, Y. (2024). Fusion-driven semi-supervised learning-based lung nodules classification with dual-discriminator and dual-generator generative adversarial network. BMC Medical Informatics and Decision Making, 24(1): 403. https://doi.org/10.1186/s12911-024-02820-9

[6] Abdulhussien, W.R., Al-Safi, J.K.S., Jwaid, W.M. (2024). Artificial intelligence-based DS-PSO algorithm for enhanced frequency response in digital IIR filters. Indonesian Journal of Electrical Engineering and Informatics (IJEEI), 12(4): 858-869. https://doi.org/10.52549/ijeei.v12i4.5885

[7] Sasmito, B., Setiadji, B.H., Isnanto, R.R. (2025). Object detection: Real-time road damage detection and geolocation using YOLOv8 and GNSS integration. Ingénierie des Systèmes d'Information, 30(9): 2321-2329. https://doi.org/10.18280/isi.300909

[8] Raju, J.K., Shankar, T. (2024). Real-time snake detection with alert systems using deep learning. Grenze International Journal of Engineering & Technology (GIJET), 10(1): 1323.

[9] Hu, F., Wang, P., Li, Y., Duan, C., et al. (2023). Watch out venomous snake species: A solution to snakeclef2023. arXiv preprint arXiv:2307.09748. https://doi.org/10.48550/arXiv.2307.09748

[10] Miyaguchi, A., Gustineli, M., Fischer, A., Lundqvist, R. (2024). Transfer learning with self-supervised vision transformers for snake identification. arXiv preprint arXiv:2407.06178. https://doi.org/10.48550/arXiv.2407.06178

[11] Wang, H., Zhang, S., Zhang, C., Liu, Z., Huang, Q., Ma, X., Jiang, Y. (2025). Snake-DETR: A lightweight and efficient model for fine-grained snake detection in complex natural environments. Scientific Reports, 15(1): 1282. https://doi.org/10.1038/s41598-024-84328-w

[12] Ahmed, K., Gad, M.A., Aboutabl, A.E. (2024). Snake species classification using deep learning techniques. Multimedia Tools and Applications, 83(12): 35117-35158. https://doi.org/10.1007/s11042-023-16773-0

[13] Xia, Y., Luo, H. (2024). YODE-FEIM: An enhanced YOLOv5s algorithm for snake detection in wild environments. International Journal of Science and Engineering Applications, 13(10): 76-81. https://doi.org/10.7753/IJSEA1310.1016

[14] Mane, V., Khot, N., Toraskar, R., Inamdar, S. (2025). Real-Time detection and classification of venomous and non venomous snakes using YOLOv8. Available at SSRN 5345043. https://doi.org/10.2139/ssrn.5345043

[15] Yang, Z., Sinnott, R. (2021). Snake detection and classification using deep learning. In Proceedings of the 54th Hawaii International Conference on System Sciences, pp. 1212-1221. https://doi.org/10.24251/HICSS.2021.148

[16] Shetty, S.V. (2025). Deep learning based snake intrusion monitoring system. International Journal of Engineering Technology Research & Management (IJETRM), 9(4):106-111. https://doi.org/10.5281/zenodo.15174253

[17] Naresh, E., Babu, J.A., Darshan, S.S., Murthy, S.V.N., Srinidhi, N.N. (2023). A novel framework for detection of harmful snakes using YOLO algorithm. SN Computer Science, 5(1): 52. https://doi.org/10.1007/s42979-023-02366-z

[18] Iguernane, M., Ouzziki, M., Es-Saady, Y., El Hajji, M., Lansari, A., Bouazza, A. (2025). Deep learning-based snake species identification for enhanced snakebite management. AI, 6(2): 21. https://doi.org/10.3390/ai6020021

[19] Reis, D., Kupec, J., Hong, J., Daoudi, A. (2023). Real-time flying object detection with YOLOv8. arXiv preprint arXiv:2305.09972. https://doi.org/10.48550/arXiv.2305.09972

[20] Wu, T., Dong, Y. (2023). YOLO-SE: Improved YOLOv8 for remote sensing object detection and recognition. Applied Sciences, 13(24): 12977. https://doi.org/10.3390/app132412977

[21] Talib, M., Al-Noori, A.H., Suad, J. (2024). YOLOv8-CAB: Improved YOLOv8 for real-time object detection. Karbala International Journal of Modern Science, 10(1): 5. https://doi.org/10.33640/2405-609X.3339

[22] Yaseen, M. (2024). What is YOLOv9: An in-depth exploration of the internal features of the next-generation object detector. arXiv preprint arXiv:2409.07813.

https://doi.org/10.48550/arXiv.2408.15857
[23] Jahan, M.K., Bhuiyan, F.I., Amin, A., Mridha, M.F., Safran, M., Alfarhood, S., Che, D. (2025). Enhancing the YOLOv8 model for realtime object detection to ensure online platform safety. Scientific Reports, 15(1): 21167. https://doi.org/10.1038/s41598-025-08413-4
[24] Albdair, M., Saihood, A., Hamad, A.M., Sahi, A. (2025). Secured multi-objective optimisation-based protocol for reliable data transmission in underwater wireless sensor networks. Mesopotamian Journal of CyberSecurity, 5(1): 216-239. https://doi.org/10.58496/MJCS/2025/015
[25] Mohsin, Z.R., Khan, F. (2025). Modelling software development effort using data-driven models. Journal of Intelligent Systems & Internet of Things, 15(2): 29-40. https://doi.org/10.54216/JISIoT.150203
[26] Najm, H., Mohammed, B.K., Naman, H.A., Al Bazar, H., Mahdi, M.S., Abdulhussien, W.R. (2025). A robust iris localization and texture extraction scheme for iris authentication systems. Journal of Advanced Research Design, 136(1): 232-242. https://doi.org/10.37934/ard.136.1.232242