International Journal of
Human Computations & Intelligence
Vol. 04, Issue. 04, 2025

RESEARCH ARTICLE OPEN ACCESS

Automated Detection of Large Animals in Road Scene Environments Using Deep Learning

Kante Murali . Rayapudi Prashanthi . J K Neelima Bai . M Chandramohan Reddy

Department of Electronics and Communication Engineering,
Narayana Engineering College, Nellore, India.

DOI: **10.5281/zenodo.15429494**
Received: 28 April 2025 / Revised: 09 May 2025 / Accepted: 16 May 2025

Abstract – Automated detection of large animals in road scenes plays a crucial role in enhancing the safety of autonomous vehicles, particularly in regions where wildlife-related accidents are common. This paper introduces a deep learning-based explanation for detecting and classifying ten large animal classes within road scene environments, such as dogs, horses, cows, and bears. A specialized dataset was fetched using selected classes from the COCO and Open Images V5 datasets, annotated in the COCO format. Four advanced object detection models were trained and evaluated with the EfficientDet-D1, RetinaNet R-50-FPN, Faster R-CNN R-50-FPN, and Cascade R-CNN R-50-FPN. Results show that RetinaNet R-50-FPN achieved the highest mean Average Precision (mAP) of 0.83 for one joint class and 0.69 for ten classes while also delivering the fastest inference speed at 50.6 FPS for one-class detection and 45.2 FPS for multi-class detection. EfficientDet-D1 achieved a mAP of 0.89 for one joint class and 0.77 for ten classes, offering competitive performance but with slightly slower inference speeds. The findings highlight RetinaNet as the most effective and efficient model for real-time large animal detection in road scenes, offering significant potential for integration into modern autonomous driving systems.

Index Terms – Automated detection, large animals, road scene environments, deep learning, object detection, RetinaNet, EfficientDet, mean Average Precision (mAP), real-time inference, wildlife safety, neural networks.

I. INTRODUCTION

For computer vision systems of uncrewed vehicles, accurately identifying large animals in images is a significant difficulty, and how frequently wild animals are involved in traffic accidents is particularly crucial [1]. The early attempts to address this issue relied on traditional computer vision techniques using hand-crafted features, such as Haar cascades, Histograms of Oriented Gradients (HOG), and Local Binary

MILESTONE RESEARCH.IN OPEN ACCESS

ISSN (Online): 2583-5696
Int. Jr. of Hum Comp. & Int.

Patterns (LBP) [2]. Although these classical methods provided a foundation, their performance proved insufficient for real-world driving scenarios' unpredictable and complex nature.

The advent of deep learning and intense convolutional neural networks (CNNs) has transformed object detection and recognition tasks across many domains, including animal detection [3]. A major benefit of deep learning models is their capacity to automatically develop hierarchical feature representations from the data, which results in more reliable and accurate detection systems [4]. A significant obstacle in training effective detection models for large animals is the rarity of such events and the scarcity of comprehensive datasets [5]. While several datasets, such as iWildCam [6], Animal Image, Oxford-IIIT Pet [7], and STL-10 [8], provide labeled images of animals, they are often limited in the number of annotated examples or the variety of species represented. However, while bigger resources like ImageNet provide enormous collections of animal images, most have no substance associated with road settings. A more promising approach is supplied by datasets such as COCO and Google's Open Images V5, which offer comprehensive annotations like segmentation masks and bounding boxes for a variety of categories, including giant animals.

Given these considerations, deep learning emerges as the most viable approach to tackle the problem of large animal detection in road scenes. This work centers attention on developing and evaluating deep learning-based methods specifically for identifying ten widespread classes of large animals “Bear," "Fox," "Dog," "Horse," "Goat," "Sheep," "Cow," "Zebra," "Elephant," and "Giraffe." In addition to detecting individual classes, a combined class detection approach is also investigated to simulate real-world scenarios where multiple species may need to be recognized collectively. To this end, a specialized dataset was constructed by curating relevant COCO and Open Images V5 images. Various state-of-the-art object detection architectures were trained and evaluated, including EfficientDet, RetinaNet (ResNet-50-FPN backbone), Faster R-CNN (ResNet-50-FPN), and Cascade R-CNN (ResNet-50-FPN). Performance was assessed using the average precision (AP) per class and the mAP across all classes, along with measuring inference times to ensure practical applicability for real-time systems.

The contribution of this work is as follows:

1. We conducted a comparative study of four state-of-the-art deep learning models (EfficientDet, RetinaNet R-50-FPN, Faster R-CNN R-50-FPN, and Cascade R-CNN R-50-FPN) to assess their effectiveness in detecting large animals in road environments.
2. A diverse dataset was created by combining selected COCO and Open Images V5 classes, including both common and rare animal species, enhancing the detector's generalization ability.
3. The models were rigorously evaluated employing mAP and frames per second (FPS) metrics for individual and combined animal classes, highlighting trade-offs between accuracy and speed.
4. The work underscores real-world applicability by addressing traffic safety by automatically detecting animals that frequently appear near roadways.

II. LITERATURE SURVEY

Parkavi et al. [9] introduced a YOLOv5-based model to detect animals on highways at night, aiming to reduce collisions and promote wildlife conservation by employing pre-processing techniques,

including CLAHE and a robust Retinex model, to improve image quality under low-light circumstances. The model was trained on various datasets and demonstrated strong performance, achieving a precision of 0.923, recall of 0.773, mAP50 of 0.802, and mAP50-95 of 0.567. Findings revealed high detection accuracy under normal night-time conditions, although challenges remain in adverse weather and dense foliage, prompting future work to improve resilience.

Sengan et al. [10] presented a DL-R-3D-YOLOv3 model for real-time object detection (ODT) on GPU-based embedded systems, focusing on moving animals. The model uses 3D motion-based deep learning to reconstruct multi-scalar 3D views to identify objects near moving vehicles better. Around 1600 images of Indian stray and wild animals, such as cats and cows, were used for training. The goal was to improve recognition using embedded vision and flexible 3D reconstruction. Despite having a lesser precision, the model produced strong positive detection results with an accuracy of around 84.18%. Simoes et al. [11] proposed a three-step object detection model to detect, classify, and count species in camera trap videos. After splitting videos into images and annotating them with MegaDetector, they extended Faster R-CNN with Inception-ResNet-v2 backbone to detect and classify 13 species. A counting method based on bounding boxes was designed. Using a French national park dataset, the model gained 73.92% mAP for classification, 96.88% mAP for detection at IoU 0.5, and 89.24% at IoU 0.75. Counting accuracy reached 87% based on detection alone and 48% when considering both detection and classification, demonstrating strong potential for wildlife monitoring.

Sato et al. [12] developed a computer vision system using machine learning to detect animals on highways, focusing on capybaras and donkeys. They trained two YOLO variants, YOLOv4 and YOLOv4-tiny, using pre-trained models. The system was evaluated on 147 images, achieving 84.87% accuracy with YOLOv4 and 79.87% with YOLOv4-tiny, demonstrating effective animal detection for road safety applications. Siddaiyan et al. [13] proposed a novel method combining LaneNet with a sliding window and YOLOv5 to detect lane lines, objects, and driveable space on major district roads (MDR). To reduce complexity, pre-processing steps like ROI extraction and bird' s-eye view transformation were applied. Using real-time data along with CULane, BDD100K, and TuSimple datasets, the model achieved 97% accuracy for lane detection and 98% for object detection. Driveable space identification showed 80%–92% precision across various datasets, validating the method's effectiveness.

Zhang et al. [14] analyzed visual landscape features of urban waterfront areas, and they predicted human perception along Xiamen’s coastal roads using linear regression and random forest models. Roads were classified into four types based on aesthetic and distinctiveness perceptions. The random forest model achieved 87% and 77% accuracy, identifying that coastal openness most positively impacted perception, while a high green visual index reduced distinctiveness. Additionally, 60.6% of the road sections showed potential for improvement.

III. METHODS & MATERIALS

This section outlines the approach for automated large animal detection in road scenes by using images from the COCO and Open Images V5 datasets, covering 10 animal classes. Annotations follow the COCO-style JSON format, including segmentation, bounding boxes, and class IDs. We implemented four deep learning models: EfficientDet, RetinaNet R-50-FPN, Faster R-CNN R-50-FPN, and Cascade R-

CNN R-50-FPN. Each model employs different detection strategies, including Focal Loss in RetinaNet and multi-stage refinement in Cascade R-CNN. All models were pre-trained on the COCO 2017 dataset and fine-tuned for our 10-class task. The training was performed using PyTorch for all models except EfficientDet, which was trained with TensorFlow/Keras. The input resolution was 640×384 pixels, and data augmentation techniques were applied to improve model robustness.

A. Dataset Description: To ensure diversity and sufficient representation of various species, we sourced images from two major open datasets: the COCO dataset [15] and Google's Open Images V5 [16]. From the COCO dataset, we selected the following animal classes: Dog, Horse, Sheep, Cow, Bear, Elephant, Zebra, and Giraffe. Although species like Elephant, Zebra, and Giraffe are uncommon in typical road environments of the target region, they were intentionally included to enhance the detector's generalization capability and robustness in recognizing a wider range of large animals. In addition, the Open Images V5 dataset contributed further diversity by providing images of Fox, Goat, and Deer classes. These additional species are relevant for real-world detection scenarios where encounters with wild or domestic animals can pose serious risks to traffic safety.

Fig. 1: Fragment of the data set

All annotations were standardized and formatted in the COCO-style JSON structure, ensuring compatibility across different detection models and frameworks. The annotations for each image include the following key fields:

- **Segmentation**: Defines the coordinates of the polygon outlining the object.
- **Area**: Measures the pixel area covered by the object.
- **IsCrowd**: Indicates whether the image contains a single object (0) or multiple overlapping instances (1).
- **BBox**: Specifies the bounding box coordinates (x, y, width, height) for the object.
- **Category_id**: Assigns an identifier representing the object's class within the common supercategory "animal."
- **ID**: Provides a unique identifier for each annotated image.

Table I presents a detailed statistical breakdown of the dataset, summarizing the number of images and bounding box annotations available for each class across the training and testing splits. Figure 1 illustrates a fragment of the dataset's annotation structure.

Table 1: Class-wise image number

Class	Training Images	Training Boxes	Testing Images	Testing Boxes
Dog	4385	5508	177	218
Horse	2941	6587	128	273
Sheep	1529	9509	65	361
Cow	1968	8147	87	380
Elephant	2143	5513	89	255
Bear	960	1294	49	71
Zebra	1916	5303	85	268
Giraffe	2546	5131	101	232
Fox	460	584	10	12
Goat	274	599	14	34
Total Number	**19122**	**48175**	**805**	**2104**

B. Methodology: To address the challenge of large animal detection in road scene environments, we explored four deep-learning architectures selected based on their proven effectiveness in object detection tasks. The models cover one- and two-stage detection paradigms, ensuring a balance between detection accuracy and inference speed.

i. **EfficientDet:** EfficientDet is a one-stage object detection framework emphasizing computational efficiency while maintaining competitive accuracy by introducing a compound scaling method that uniformly scales the network's resolution, depth, and width [17]. EfficientDet is built upon EfficientNet as its backbone and integrates a BiFPN (Bidirectional Feature Pyramid Network) for effective multi-scale feature fusion. EfficientDet was trained with an initial learning rate of 0.01, using input images resized to 640×384 pixels, and a batch size of 8. The object detection loss function for EfficientDet is a weighted sum of classification loss and bounding box regression loss:

$$L = \lambda_{cls}.L_{cls} + \lambda_{reg}.L_{reg}$$

where, L_{cls} is the classification loss, L_{reg} is the bounding box regression loss, λ_{cls} and λ_{reg} are balancing coefficients.

ii. **RetinaNet R-50-FPN:** RetinaNet is a pioneering one-stage detector designed to address the severe class imbalance between foreground and background during training [18]. Its hallmark innovation is the Focal Loss, which down-weights the loss assigned to well-classified examples, allowing the model to focus on harder, misclassified instances. The overall RetinaNet loss is expressed as:

$$L = L_{focal} + L_{reg}$$

where, L_{focal} is the classification loss, and L_{reg} is the bounding box regression loss (smooth L1 loss).

The focal loss formulation is:

$$L_{focal}(P_t) = - \alpha_t (1 - P_t)^{\gamma} \log (p_t)$$

This architecture utilizes a ResNet-50 backbone with a Feature Pyramid Network (FPN) to detect objects at different scales efficiently. Like EfficientDet, training was initiated using weights pre-trained on the COCO 2017 dataset after adjusting the network heads to match our custom 10-class animal detection task.

iii. **Faster R-CNN R-50-FPN:** Faster R-CNN is a two-stage object detector that integrates a Region Proposal Network (RPN) to generate candidate object proposals directly, eliminating the need for external proposal algorithms like selective search [19]. The first stage proposes regions, and the second stage classifies and refines these proposals.

The total loss for Faster R-CNN can be expressed as:

$$L = L_{rpn\_cls} + L_{rpn\_reg} + L_{roi\_cls} + L_{roi\_reg}$$

where, L_{rpn_cls} is the classification loss for region proposals, L_{rpn_reg} is the bounding box regression loss for proposals, L_{roi_cls} is the classification loss for final detections, and L_{roi_reg} is the bounding box regression loss for final detections.

The RPN and the final detection heads employ smooth L1 loss for localization and cross-entropy loss for classification.

Faster R-CNN uses a ResNet-50 backbone with FPN to enhance feature extraction across scales, providing a strong balance between detection accuracy and processing efficiency.

iv. **Cascade R-CNN R-50-FPN:** Cascade R-CNN extends the Faster R-CNN architecture by introducing multiple sequential detection stages [20]. Each subsequent detector is trained with progressively stricter IoU thresholds, improving the localization quality and reducing overfitting to easy examples. The overall loss for Cascade R-CNN is a cumulative sum of losses at each cascade stage:

$$\mathrm{L} = \sum_{i=1}^{N} (L_{cls}^{(i)} + L_{reg}^{(i)})$$

Where, N is the number of stages (typically three), and $L_{cls}^{(i)}$ and $L_{reg}^{(i)}$ are the classification and regression losses at the i-th stage.

Each stage applies stricter positive sample definitions by increasing the IoU threshold, thus progressively refining object detection. Cascade R-CNN also adopts a ResNet-50-FPN backbone for feature extraction, ensuring multi-scale robustness. Figure 2 graphically represents the architecture of the Cascade R-CNN R-50-FPN neural network.

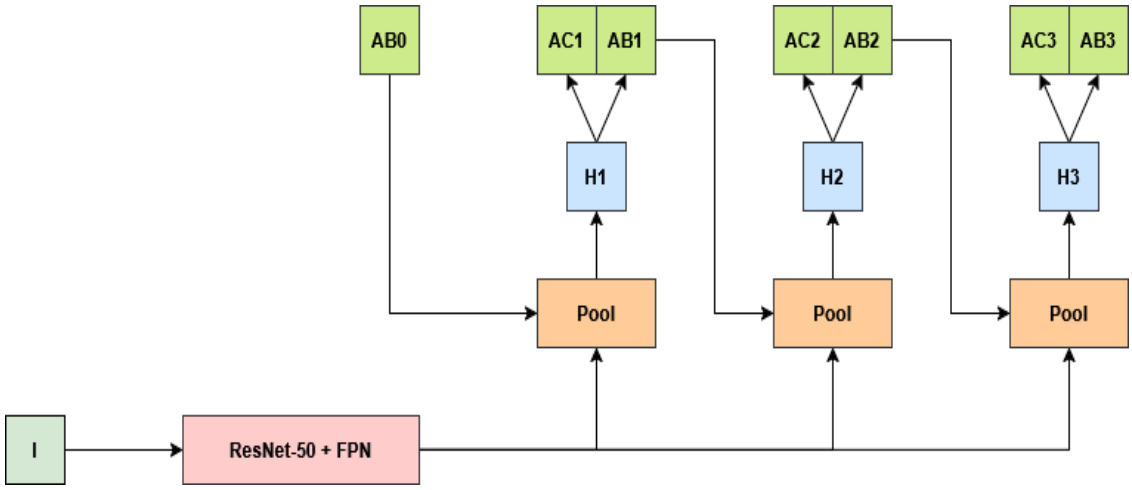

Fig. 2: Architecture of Cascade R-CNN R-50-FPN neural network

C. Training Strategy: All models, except EfficientDet, were trained using the PyTorch framework. EfficientDet training was performed using TensorFlow/Keras due to better integration with its original implementation. Table 2 exhibits the training Configuration Parameters. Pre-trained weights were utilized for initialization:

- For EfficientDet, the backbone EfficientNet weights were loaded.
- For RetinaNet, Faster R-CNN, and Cascade R-CNN, full networks pre-trained on COCO 2017 were adapted by reinitializing the final prediction layers to match our 10 animal classes.

Table 2: Training Configuration Parameters for Deep Learning Models

Parameter	Value
Input Resolution	640x384x3
Batch Size	8 images
Initial Learning Rate	0.01
Backbone Freezing	Early-stage freezing only for EfficientDet
Data Augmentation	Horizontal flips, Random crops

IV. RESULT & DISCUSSION

This section will discuss the experimental setup, detection performance, and inference efficiency of the selected deep-learning models. RetinaNet R-50-FPN demonstrated the best balance of speed and accuracy, making it ideal for real-time animal detection. EfficientDet-D1 offered decent performance but was slightly slower. Cascade and Faster R-CNN achieved high precision but lacked the inference speed required for real-time deployment in road scene environments.

A. Experimental Setup:

The experimental evaluations were conducted on a high-performance HP desktop, and the system was equipped with an NVIDIA Tesla V100 GPU featuring 32 GB of VRAM, leveraging CUDA technology for accelerated computation of matrix operations and neural network training tasks. A powerful Intel Xeon Gold 6154 CPU with 16 physical cores running at 3.00 GHz managed data preprocessing, loading, and auxiliary tasks, ensuring CPU operations were not bottlenecks during GPU-intensive processes. Additionally, the system included 128 GB of RAM, which allowed efficient handling of large datasets, batch loading, and extensive data augmentation pipelines without memory limitations. The software environment was set up with Ubuntu 20.04 LTS as the operating system, CUDA Toolkit 11.2, PyTorch 1.10 for training RetinaNet, Faster R-CNN, and Cascade R-CNN architectures, and TensorFlow 2.7 (with Keras API) for EfficientDet models. COCO API tools were utilized to handle dataset annotation, while other libraries like OpenCV and Albumentations were utilized for picture preprocessing and augmentation activities. By entirely using the V100's Tensor Cores through mixed-precision training, training speed was significantly increased while numerical stability was maintained. All models used the same hyperparameters, such as batch size, input resolution, and learning rate scheduling, to provide a reliable and equitable comparison.

B. *Comparative Performance Analysis of Detection Models:*

The comparative evaluation of object detection performance across the four deep learning architectures reveals significant insights into their suitability for detecting large animals in road scene environments. As shown in Table 3, EfficientDet-D1 consistently outperforms the other models across most animal classes, achieving the highest mean Average Precision (mAP) of 0.77. This suggests its effectiveness in balancing detection quality and computational efficiency, especially in detecting high-risk animals, such as dogs, horses, and cows, most commonly involved in road incidents. EfficientDet achieved an AP of 0.89 for dogs and 0.86 for horses**,** outperforming all other architectures in these critical categories.

Table 3: Quality of the big animal detection of the testing sample

Class	Cascade R-CNN R-50-FPN	Faster R-CNN R-50-FPN	RetinaNet R-50-FPN	EfficientDet-D1
AP_{dog}	0.81	0.81	0.83	0.89
AP_{horse}	0.75	0.76	0.77	0.86
AP_{sheep}	0.68	0.67	0.65	0.73
AP_{cow}	0.65	0.66	0.60	0.78
$AP_{elephant}$	0.82	0.83	0.84	0.90
AP_{bear}	0.81	0.87	0.89	0.93
AP_{zebra}	0.84	0.88	0.88	0.90
$AP_{giraffe}$	0.87	0.86	0.87	0.89
AP_{fox}	0.21	0.18	0.19	0.23
AP_{goat}	0.39	0.44	0.41	0.55
Quality Metric				
mAP	**0.68**	**0.70**	**0.69**	**0.77**

The Cascade R-CNN and Faster R-CNN models also showed competitive performance, particularly on more visually distinct classes like giraffe, zebra, and elephant, where their multi-stage detection approach helped refine bounding boxes effectively. Cascade R-CNN achieved a strong AP of 0.87 for giraffes and 0.84 for zebras, closely matching EfficientDet's results. However, their lower performance in smaller or less frequently represented classes, such as fox and goat, indicates challenges in handling underrepresented or ambiguous animal categories. While generally solid, RetinaNet showed slightly less consistency than other models, particularly in the cow and sheep categories.

This may be attributed to the one-stage architecture's sensitivity to class imbalance and occlusion, which are common in real-world datasets. The lower AP scores for fox and goat across all models reflect a dataset-driven limitation: fewer annotated samples for these species reduce the model's generalization ability. Nevertheless, EfficientDet maintains a somewhat greater accuracy even in these classes, demonstrating its robust feature scaling and compound backbone-head design. Figure 3 displays the deep learning model's performance of the mAP quality metric. To sum up, EfficientDet is an excellent choice for real-time large animal detection systems in road environments since it provides exceptional precision and deployment efficiency.

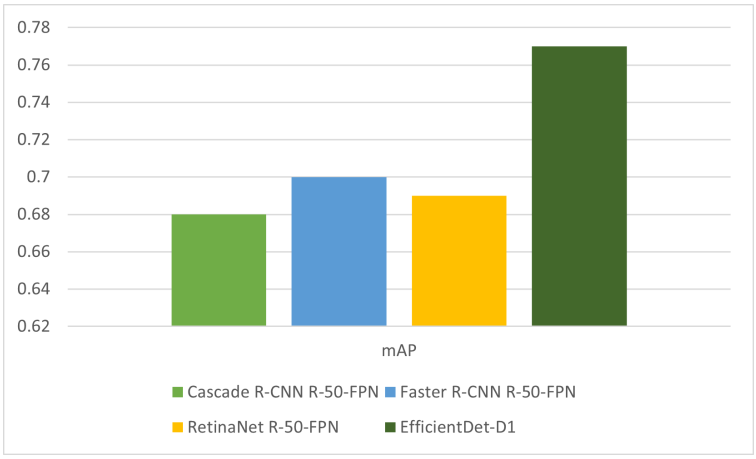

Fig. 3: The mAP performance of the deep learning models

C. Unified Class Detection Performance Analysis: When considering the unified detection of all large animal classes as a single object category, the performance metrics underscore each evaluated architecture's strengths and trade-offs. As indicated in Table 4, EfficientDet-D1 achieved the highest mAP of 0.91, clearly outperforming the other three models.

Table 4: Performance Metrics for Unified Animal Detection Class on Testing Set

Quality Metric	Cascade R-CNN R-50-FPN	Faster R-CNN R-50-FPN	RetinaNet R-50-FPN	EfficientDet-D1
mAP (One-Class)	0.84	0.85	0.86	0.91

The competing architectures RetinaNet, Faster R-CNN, and Cascade R-CNN also demonstrated competent performance, with mAP scores between 0.84 and 0.86. These results suggest that traditional region-based detectors perform reliably under a joint-class scenario, mainly due to their refined region proposal mechanisms and strong feature hierarchies. RetinaNet's focal loss function contributed to balanced learning despite class variations, enabling it to achieve slightly higher accuracy among the three.

However, EfficientDet's more advanced compound scaling and lightweight backbone allowed it to learn more generalized representations across all species, which is particularly useful when real-time or edge-computing requirements constrain model deployment. The significant boost in its mAP under a unified label also implies that EfficientDet can be an effective end-to-end detection tool in practical road safety systems where quick, reliable identification of large animals, regardless of species, is essential.

D. Real-Time Detection Efficiency and Computational Throughput:

The efficiency of object detection models in real-time scenarios is as critical as their accuracy, particularly in automated driving systems where delays can lead to hazardous consequences. Table 5 compares the detection speeds of several state-of-the-art architectures. The fastest model was RetinaNet R-50-FPN, which produced 50.6 FPS for one-class detection and 45.2 FPS for identifying all ten animal classes. Because of its competitive precision and speedy performance, it is a good choice for systems that require real-time responsiveness.

Table 5: Inference Speed Comparison for One-Class and Multi-Class Animal Detection

Performance Metric	Cascade R-CNN R-50-FPN	Faster R-CNN R-50-FPN	RetinaNet R-50-FPN	EfficientDet-D1
FPS (One-Class Detection)	26.5	39.0	50.6	47.2
FPS (10-Class Detection)	25.1	37.6	45.2	43.5

EfficientDet-D1 also showed commendable performance with 47.2 FPS and 43.5 FPS for one-class and multi-class detection, respectively. While slightly trailing RetinaNet in speed, it maintains a favorable balance between throughput and detection quality, especially on hardware-constrained systems. On the other hand, Cascade R-CNN was the slowest on both tasks because of its multi-stage design, while Faster R-CNN provided a middling pace. This emphasizes how speed and model complexity must be balanced, making RetinaNet a good choice for situations where processing time is just as important as accuracy. To sum up, RetinaNet R-50-FPN is the most useful architecture for large animal recognition in real-time, providing speed and efficiency essential for maintaining safety in road scene settings.

IV. CONCLUSION AND FUTURE WORK

In this study, we designed an automated system for detecting large animals in road scene environments employing deep learning models, including EfficientDet-D1, RetinaNet R-50-FPN, Faster R-CNN R-50-FPN, and Cascade R-CNN R-50-FPN. These models were trained on a custom dataset created by combining COCO and Open Images V5 images annotated for relevant animal classes. The findings show that using deep learning techniques to improve road safety in autonomous driving systems, especially identifying huge animals that might endanger moving vehicles, is feasible. Despite the promising outcomes, there are several areas for improvement. To improve generalization in practical applications, future research should concentrate on expanding the dataset to include other animal classifications and a variety of road settings. For real-time detection in autonomous cars, these models must be optimized for deployment on edge devices with constrained processing power. Investigating the combination of many sensor modalities, including radar and lidar, may also improve detection robustness and accuracy, especially under difficult circumstances like bad weather or limited visibility.

REFERENCES

1. Anjali, K., Reddy, D. C., Prasanthi, B., Hussain, S. M., L. V., & Budithi, R. B. (2025). A comprehensive survey of technologies for wildlife detection and accident prevention. *Proceedings of the 2025 3rd International Conference on Intelligent Systems, Advanced Computing and Communication (ISACC)*, 823–830. IEEE.
2. Majeed, H. L., Hassen, O. A., Farhan, D. A., Gromov, Y. Y., Sheoran, K., & Dhand, G. (2025). Computer vision of smile detection based on machine and deep learning approach. *Journal of Cybersecurity and Information Management (JCIM)*, 16(1), 208–230.
3. Madapudi, R. K., Rao, A. A., & Merugu, G. (2012). Change requests artifacts to assess impact on structural design of SDLC phases. *International Journal of Computer Applications*, 54(18), 21–26.
4. Reddy, K. U. K., Shabbiha, S., & Kumar, M. R. (2020). Design of high security smart health care monitoring system using IoT. *International Journal*, 8.
5. Seshakagari, H. R. B., et al. (2025). Dynamic financial sentiment analysis and market forecasting through large language models. *International Journal of Human Computations & Intelligence*, 4(1), 397–410.

6. Beery, S., Agarwal, A., Cole, E., & Birodkar, V. (2021). The iWildCam 2021 competition dataset. *arXiv preprint*, arXiv:2105.03494.
7. Hong, J., Huang, K., Liang, H.-N., Wang, X., & Zhang, R. (2020). Fine-grained image classification with object-part model. In *Advances in Brain Inspired Cognitive Systems: Proceedings of the 10th International Conference (BICS 2019)* (pp. 233–243). Springer.
8. Singh, H., Swagatika, S., Venkat, R. S., & Saxena, S. (2019). Justification of STL-10 dataset using a competent CNN model trained on CIFAR-10. In *2019 3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA)* (pp. 1254–1257). IEEE.
9. Parkavi, K., Ganguly, A., Banerjee, A., Sharma, S., & Kejriwal, K. (2025). Enhancing road safety: Detection of animals on highways during night. *IEEE Access*.
10. Sengan, S., Kotecha, K., Vairavasundaram, I., Velayutham, P., Varadarajan, V., Ravi, L., & Vairavasundaram, S. (2021). Real-time automatic investigation of Indian roadway animals by 3D reconstruction detection using deep learning for R-3D-YOLOv3 image classification and filtering. *Electronics*, 10(24), 3079.
11. Simões, F., Bouveyron, C., & Precioso, F. (2023). DeepWild: Wildlife identification, localisation and estimation on camera trap videos using deep learning. *Ecological Informatics*, 75, 102095.
12. Sato, D., Zanella, A. J., & Costa, E. J. X. (2021). Computational classification of animals for a highway detection system. *Brazilian Journal of Veterinary Research and Animal Science*, 58(Special), 1–10.
13. Siddaiyan, J., & Poonusamy, K. (2025). Identify the driving space for vehicle movement by lane line and road object detection using deep learning technique. *Journal of New Materials for Electrochemical Systems*, 28(1).
14. Zhang, Y., Xiong, X., Yang, S., Zhang, Q., Chi, M., Wen, X., Zhang, X., & Wang, J. (2025). Enhancing the visual environment of urban coastal roads through deep learning analysis of street-view images: A perspective of aesthetic and distinctiveness. *PLOS ONE*, 20(1), e0317585.
15. Jain, S., Dash, S., & Deorari, R. (2022). Object detection using COCO dataset. In *2022 International Conference on Cyber Resilience (ICCR)* (pp. 1–4). IEEE.
16. Krylov, I., Nosov, S., & Sovrasov, V. (2021). Open Images V5 text annotation and yet another mask text spotter. In *Asian Conference on Machine Learning* (pp. 379–389). PMLR.
17. Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10781–10790).
18. Wang, Y., Liu, W., & Xing, W. (2021). Balanced-RetinaNet: Solving the imbalanced problems in object detection. *Journal of Electronic Imaging*, 30(3), 033009.
19. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 28.
20. Fathima, A. S., Basha, S. M., Ahmed, S. T., Khan, S. B., Asiri, F., Basheer, S., & Shukla, M. (2025). Empowering consumer healthcare through sensor-rich devices using federated learning for secure resource recommendation. *IEEE Transactions on Consumer Electronics*.
21. Cai, Z., & Vasconcelos, N. (2019). Cascade R-CNN: High quality object detection and instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(5), 1483–1498.
22. Pasha, A., Ahmed, S. T., Painam, R. K., Mathivanan, S. K., Mallik, S., & Qin, H. (2024). Leveraging ANFIS with Adam and PSO optimizers for Parkinson's disease. *Heliyon*, 10(9).
23. Kumar, S. S., Ahmed, S. T., Sandeep, S., Madheswaran, M., & Basha, S. M. (2022). Unstructured oncological image cluster identification using improved unsupervised clustering techniques. *Computers, Materials & Continua*, 72(1).