Cat Breed Classification with YOLOv11 and Optimized Training

Hafedh Mahmoud Zayani
Department of Electrical Engineering, College of Engineering, Northern Border University, Arar, Saudi Arabia
hafedh.zayani@nbu.edu.sa

Amani Kachoukh
Department of Information Systems, Faculty of Computing and Information Technology, Northern Border University, Rafha, Saudi Arabia
amani.khasookh@nbu.edu.sa

Refka Ghodhbani
Center for Scientific Research and Entrepreneurship, Northern Border University, 73213, Arar, Saudi Arabia
refka.ghodhbani@nbu.edu.sa

Nouha Khediri
Department of Computer Sciences-Faculty of Computing and Information Technology, Northern Border University, Rafha, Saudi Arabia
nuha.khediri@nbu.edu.sa

Eman H. Abd-Elkawy
Department of Computer Science, Faculty of Computing and Information Technology, Northern Border University, Rafha, Saudi Arabia | Department of Mathematics and Computer Science, Faculty of Science, Beni-Suef University, Beni-Suef, Egypt
eman.hassan@nbu.edu.sa

Ikhlass Ammar
Computer Science, Faculty of Sciences of Tunis (FST), University of Tunis El Manar, Tunisia | OASIS Laboratory, National Engineering School of Tunis, University of Tunis El Manar, Tunisia
ikhlass_ammar@yahoo.fr

Marouan Kouki
Department of Information Systems, Faculty of Computing and Information Technology, Northern Border University, Rafha, Saudi Arabia
marouan.kouki@nbu.edu.sa

Taoufik Saidani
Center for Scientific Research and Entrepreneurship, Northern Border University, 73213, Arar, Saudi Arabia
taoufik.saidan@nbu.edu.sa (corresponding author)

Received: 12 January 2025 | Revised: 3 February 2025 and 7 February 2025 | Accepted: 8 February 2025

 DOI: https://doi.org/10.48084/etasr.10218

ABSTRACT

Accurate identification of cat breeds poses a significant challenge due to subtle inter-breed differences and intra-breed variability. This study leverages YOLOv11, the latest version of the YOLO family, to address these challenges through advanced deep-learning techniques. By training on a dataset consisting of five distinct cat breeds (Persian, Maine Coon, Siamese, Pallas's Cat, and Bengal), the model demonstrates exceptional capability in identifying nuanced breed-specific features. Data augmentation techniques were employed to enhance the dataset's diversity, while various optimization algorithms (Adam, Adamax, NAdam, AdamW, RAdam, RMSProp, and SGD) were evaluated to optimize the performance of the model. Experimental results showed that RAdam and SGD emerged as the top-performing optimizers, achieving an average recall of 96.8%, precision of 97.2%, and mAP50 of 98.1%, significantly outperforming other optimization methods. In contrast, RMSProp exhibited the lowest performance, particularly in terms of precision and mean Average Precision (mAP50). Additionally, data augmentation techniques were applied to enhance the diversity of the dataset, improving the robustness of the model. These findings highlight the effectiveness of YOLOv11 in cat breed classification, with potential applications in pet identification, animal conservation, and veterinary diagnostics.

Keywords-cat breed classification; deep learning; YOLOv11; optimizer

I. INTRODUCTION

Accurate cat breed identification is a challenging task due to subtle variations between breeds. While many cat breeds share similar body structures, distinguishing between them requires careful attention to features such as coat patterns, facial structure, and body size. This challenge is further compounded by the intra-breed variability, where individuals within a breed can exhibit significant differences in appearance. This study proposes a deep learning-based approach to address the intra-breed variability challenge. Five breeds were selected to represent a spectrum from highly domesticated (Persian and Maine Coon) to wild-like (Pallas's Cat and Bengal), with Siamese as an intermediate.

Deep learning techniques have driven significant advances in object detection, with YOLO emerging as a leading framework. YOLO's latest version, YOLOv11, released in September 2024 [1], builds on the success of its predecessors, introducing significant improvements in accuracy and speed. This versatile model can be applied to various tasks, including object detection, segmentation, classification, and estimation. To achieve these advances, YOLOv11 leverages state-of-the-art deep learning techniques, such as advanced neural network architectures and optimization algorithms. This study employs YOLOv11 to accurately classify cat images after data augmentation. This model is well-suited for object detection tasks, including image classification. Training the model on a diverse dataset of cat images and experimenting with various optimizers (Adam, Adamx, NAdam, AdamW, RAdam, RMSProp, and SGD) aims to capture the subtle nuances between breeds, achieve high accuracy in breed identification, and select the optimizer that yields the highest performance in terms of recall, precision, and mean Average Precision (mAP).

In recent years, the classification of cat breeds has become a new field of study. Numerous researchers have looked into different methods for correctly identifying distinct cat breeds from images. Two conventional techniques have been used for this problem, including Convolutional Neural Networks (CNN) and Support Vector Machines (SVM). However, these techniques frequently fail to capture the tiny differences between cat breeds, particularly when it comes to subtle variances in shape, coat patterns, and facial features. In [2], a CNN approach was used to classify multiple cat breeds within a single image. The proposed method utilized a multi-object detection and classification pipeline, capable of identifying and categorizing multiple cats simultaneously.

Although object detection has been the main application for YOLO, current research has shown that it is also useful for image classification tasks. Researchers have successfully used YOLO to classify cat breeds by utilizing its ability to locate and categorize things within an image. In [3], a novel approach was proposed to detect cat breeds and emotions using a combination of YOLO, CNN, and Canny edge detection. YOLO is utilized to identify and localize cats within images, while CNNs extract and classify facial features for emotion recognition. Canny edge detection is employed to enhance facial features for more accurate emotion classification. This hybrid approach aims to improve the accuracy of cat breed and emotion detection, particularly in challenging scenarios with varying lighting conditions and occlusions. The study in [4] focused on developing a system to monitor the behavior of pet cats using a YOLO model and a Raspberry Pi. The YOLO model was employed to detect and track the cat's movements and actions within a designated area. By analyzing the cat's behavior patterns, the system can provide insight into the cat's overall well-being, identify potential health issues, and assist in training and behavioral correction.

Several studies have focused on the detection of breeds of cats or dogs, such as YOLOv5 [5] or YOLOv8 [6]. In [7], a cat nose detection approach was proposed using YOLO and the scale-invariant feature transform. YOLO has been successfully applied to various object detection tasks, including human detection, vehicle detection, and identification of animal species. This algorithm has been used in many cases and versions [8-11]. This study employed the latest version, YOLOv11, to improve cat breed detection. The model's performance was further enhanced through the optimization process, which involves experimenting with various optimization algorithms, aiming to enhance the understanding of cat breed classification.

II. YOLOV11 MODEL

This section introduces the YOLOv11 architecture and compares it with some previous versions [13]. Figure 1 shows the architecture of YOLOv11 that optimizes both speed and accuracy compared to the previous versions. Its architecture consists of three blocks: Backbone, Neck, and Head. To highlight the novel contributions of YOLOv11, this section focuses on the updated modules compared to its predecessors. A concise overview of these advances aims to provide a clear understanding of the specific improvements that have led to enhanced performance and efficiency in object detection tasks.

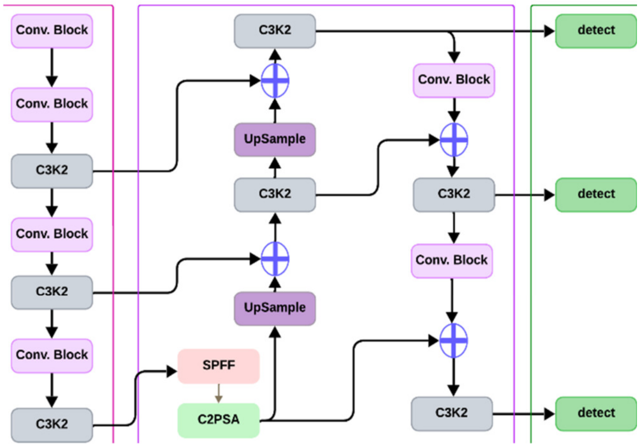

Fig. 1. YOLOv11 architecture.

A. C3K2 module

The C3K2 module is a core component of the backbone block, designed to optimize information flow and computational efficiency. It achieves this by splitting the feature map into smaller segments and applying a series of 3×3 kernel convolutions. This approach reduces computational cost while preserving the model's ability to extract crucial features. The C3K2 module offers improved feature representation compared to the C2f module in YOLOv8, while requiring fewer parameters. Additionally, the C3K2 module incorporates the C3K module, which shares a structure similar to that of the C2F module but without the splitting operation. These modules are shown in Figure 2. The design of C3K2 helps to maintain a balance between speed and accuracy, leveraging the benefits of the CSP structure.

B. SPFF Module

The Spatial Pyramid Pooling Fast (SPFF) module, shown in Figure 3, is designed to pool features from different regions of an image at varying scales to capture objects of different sizes, especially small objects. The module is used in YOLOv9, but YOLOv11 further developed it to pick up small objects. This module ensures that the last version of YOLO can maintain real-time speed to detect objects across diverse scales.

C. C2PSA Module

The Cross Stage Partial with Spatial Attention (C2PSA) module is a novel component in YOLOv11 that incorporates attention mechanisms. This component enhances the model's ability to focus on significant image regions, particularly smaller or partially obscured objects, by prioritizing spatially relevant parameters within the feature maps.

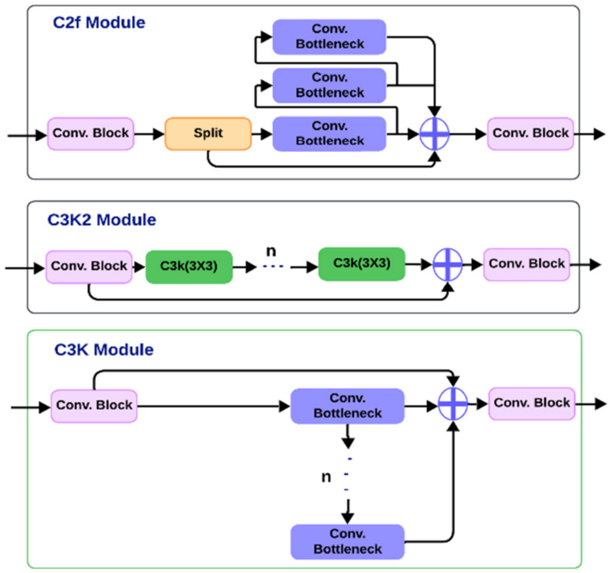

Fig. 2. C2F, C3K2, and C3K modules' architectures.

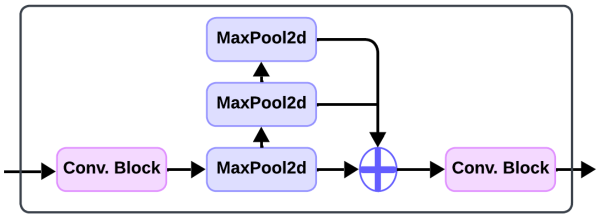

Fig. 3. SPFF module.

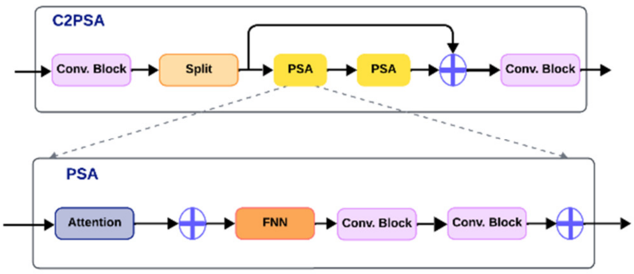

Fig. 4. C2PSA block architecture.

The C2PSA module leverages two Partial Spatial Attention (PSA) modules to enhance feature extraction and processing, as shown in Figure 4. Each PSA module applies position-sensitive attention and feed-forward networks to separate branches of the feature map. By concatenating the outputs of these modules, the C2PSA module refines the model's ability to focus on important regions within the image selectively. This spatial attention mechanism allows YOLOv11 to excel in scenarios that require precise detection of fine object details, surpassing the performance of previous versions such as YOLOv8.

III. OPTIMIZERS

Optimizers are algorithms used to adjust the parameters of a neural network during training to minimize the loss function. Some popular optimizers are:

- Adam: Combines the advantages of RMSprop and Momentum, adapting the learning rate for each parameter. It is widely used and often a good default choice.
- Adamax: A variant of Adam, using the infinity norm instead of the L2 norm. This optimizer exhibits greater stability than Adam, particularly when dealing with sparse gradients.
- NAdam: Combines the benefits of Adam and Nesterov Momentum, accelerating convergence. It can be more sensitive in hyperparameter tuning.
- AdamW: Adam with Weight Decay, incorporating weight decay to prevent overfitting and regularize the model.
- RAdam: Rectified Adam, using a rectified learning rate to improve stability and convergence.
- RMSprop: Root Mean Square Propagation, adapting the learning rate for each parameter, which is effective for dealing with noisy gradients.
- Stochastic Gradient Descent (SGD): A basic optimization algorithm that updates parameters using the gradient of the loss function. It can be slow to converge, but it can find global minima.

The choice of the optimizer depends on various factors, including the specific problem, dataset size, and model architecture. Experimenting with different optimizers and hyper-parameters is often beneficial to find the best configuration for training a dataset.

IV. METHODOLOGY

YOLOv11 was designed to be faster and more accurate than its predecessors. The following steps were involved in training the models on the cat dataset.

Fig. 5. Number of classes and images used in the cat dataset.

A. *Collect and Annotate Data*

The first step aimed at collecting images to form a dataset, which was then annotated. A custom dataset was meticulously collected, consisting of 266 images across 5 different cat breeds, as shown in Figure 5. The dataset was collected from multiple open-access sources, including Google Images, ensuring diversity in breed representation, image quality, and environmental conditions. Each image was manually annotated using the Roboflow tool, labeling breeds such as Persian, Maine Coon, Siamese, Pallas's Cat, and Bengal. Table I provides an overview of the dataset distribution, including the number of images per breed.

TABLE I. DISTRIBUTION OF IMAGES IN THE DATASET

	Persian	**Maine Coon**	**Siamese**	**Pallas**	**Bengal**
Number of images	50	53	53	55	55

B. *Data Augmentation*

The second step aimed to enrich the dataset by applying data augmentation techniques, such as rotations, flips, exposure adjustments, and noise addition, as shown in Figure 6. This process increases the diversity of the training data.

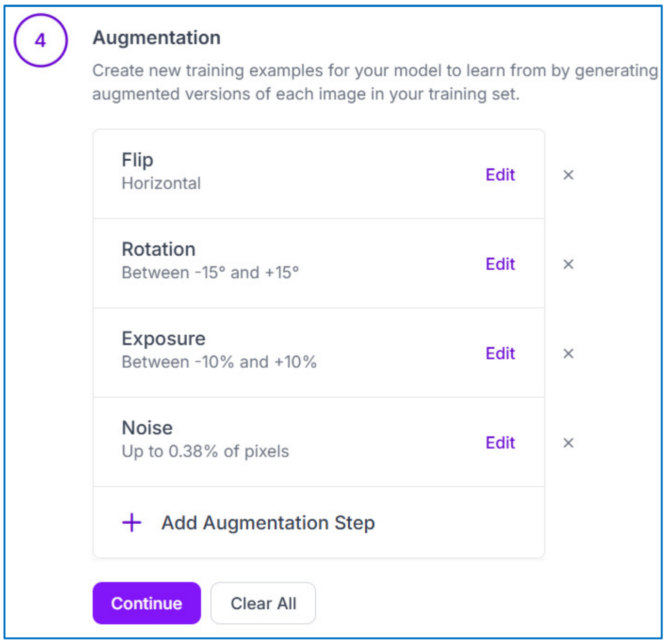

Fig. 6. Data augmentation.

C. *Selection and Configuration of the YOLO Model*

In the third step, the YOLOv11 model was selected and its hyperparameters, such as the number of output classes (corresponding to the cat genres), input image size, and anchor box dimensions were adjusted.

D. *Model Training*

After preparing the dataset, the selected model was trained. Training involves using deep learning techniques to classify images based on specific patterns and characteristics unique to each class. Training was carried out on Google Colab, where hyperparameters such as batch size, input image size, number of epochs, and optimizer type were fine-tuned. Training was carried out using a batch size of 16 and input images resized to 640 pixels. Google Colab offers a flexible platform with various hardware options, including CPUs, GPUs, and TPUs, each providing a 12-hour continuous execution window. A high-performance 12GB NVIDIA Tesla T4 GPU was selected for the experiments. The training progress was monitored to track the model's performance on a validation set using mAP, Precision, and Recall.

V. RESULTS AND INTERPRETATIONS

This study utilized a dataset of 640 cat images, divided into training (560 images), validation (54 images), and testing (25 images) sets for five distinct cat breeds. The model was trained for 50 epochs using the following optimizers: Adam, Adamax, NAdam, AdamW, RAdam, RMSProp, and SGD. YOLOv11 demonstrated impressive performance after training on the image dataset. To evaluate the efficiency and effectiveness of these optimizers, a comprehensive analysis of their performance metrics, including training time, inference time, post-processing time, Recall, Precision, and mAP50, was performed. Figure 7 illustrates the training time in hours for various optimizers, revealing that NAdam was the fastest, completing training in approximately 0.22 hours. Figure 8 shows that RMSProp exhibited the highest total processing time due to its longer post-processing phase, while NAdam demonstrated the lowest total processing time due to its shorter post-processing duration. It's worth noting that all optimizers shared a consistent pre-processing time of 0.2 ms, except for RMSProp of 0.4 ms.

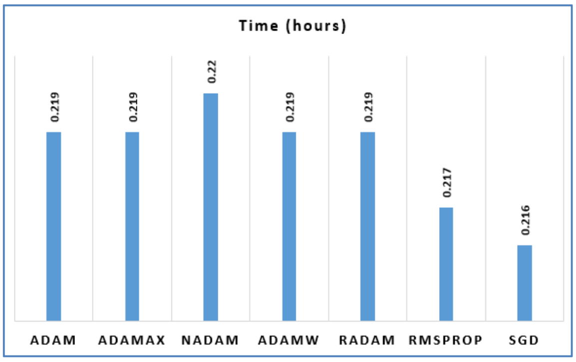

Fig. 7. Training time.

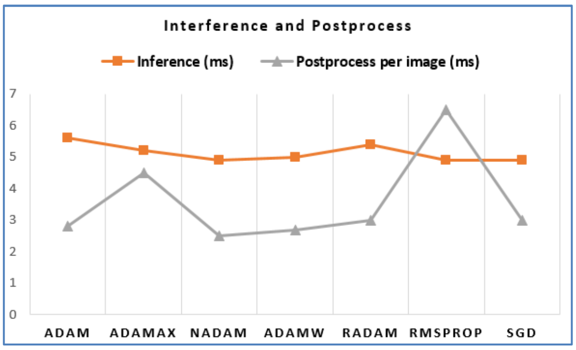

Fig. 8. Interference and postprocess per image.

Recall is a metric to assess how well a model identifies positive instances. In this case, it measures how well the model can correctly identify each cat breed. Figure 9 shows the recall for each cat breed using different optimizers. Most optimizers exhibited strong performance, with similar recall values across most breeds. In particular, SGD exhibited higher recall performance for most breeds, except Siamese and Bengal. On the contrary, Adamax achieved 100% recall for Siamese, while NAdam achieved 100% recall for Bengal.

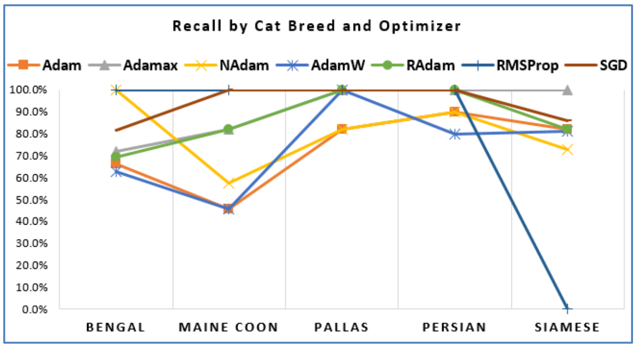

Fig. 9. Recall performance.

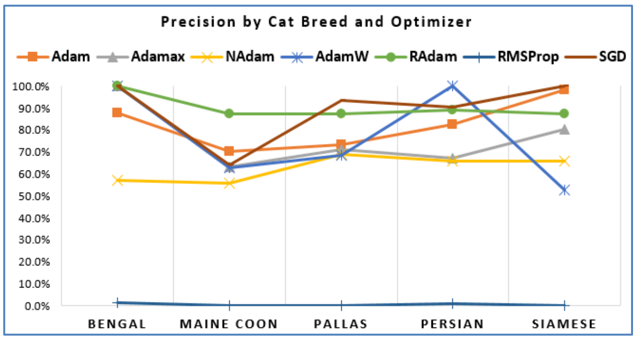

Fig. 10. Precision performance.

Precision measures the accuracy of positive predictions made by the model. In this context, it assesses how accurately the model can identify each cat breed. Figure 10 illustrates the precision for each cat breed for different optimizers. The model's performance varied across breeds, with difficulties in accurately identifying the Siamese breed, especially when using the RMSProp optimizer. RAdam and SGD exhibited higher precision compared to the other optimizers. RMSProp consistently showed the lowest precision value, indicating the weakest average performance among all optimizers.

Figure 11 presents the mAP at an Intersection Over Union (IoU) threshold of 50% (mAP50) for different cat breeds using various optimizers. Performance varied across different breeds. For example, the model using the RMSProp optimizer struggled to accurately identify all cat breeds. Adamax, RAdam, and SGD showed high mAP50 values, indicating strong average performance. Adam, Nadam, and AdamW showed slightly lower mAP50 values compared to the top-performing optimizers, suggesting a slightly weaker ability to accurately identify all cat breeds. RMSProp demonstrated a significantly lower mAP50 value, indicating the weakest average performance among all optimizers.

Based on the experimental results, RAdam and SGD demonstrated strong performance across various metrics, including the average recall, precision, and mAP50, as shown in Table II. However, Adam and Adamax exhibited slightly lower performance. Additionally, NAdam, AdamW, and RMSProp showed lower performance, particularly in terms of precision. Overall, the choice of optimizer significantly affects the performance of the cat breed classification model. For optimal performance, it is recommended to consider using optimizers such as RAdam or SGD. When examining the average recall across all classes, Adam exhibited slightly lower recall values compared to other optimizers, suggesting a slightly weaker ability to correctly identify all cat breeds. In contrast, SGD demonstrated the highest recall value, indicating the strongest overall performance among the tested optimizers.

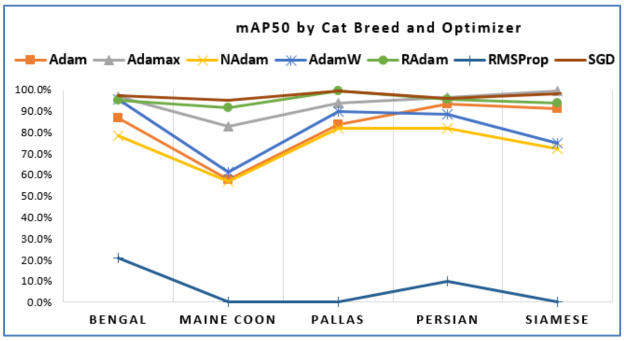

Fig. 11. Mean Average Precision (mAP50) comparison.

TABLE II. RECALL, PRECISION, AND MAP50 FOR EACH OPTIMIZER IN THE YOLOV11

Optimizers	**Recall**	**Precision**	**mAP50**
Adam	73.1%<80%	82.5%≥80%	82.5%≥80%
Adamx	90.7%≥80%	76.5%<80%	93.9%≥80%
NAdam	80.4%≥80%	62.7%<80%	74.3%<80%
AdamW	73.9%<80%	76.8%<80%	82.0%≥80%
RAdam	86.6%≥80%	90.3%≥80%	95.9%≥80%
RMSProp	80.0%≥80%	0.5%<80%	6.2%<80%
SGD	93.5%≥80%	89.6%≥80%	97.2%≥80%

VI. CONCLUSION

This study successfully demonstrated the effectiveness of YOLOv11 in accurately classifying cat breeds, even amidst subtle intra-breed variations. By leveraging a dataset of 640 annotated images and employing data augmentation techniques, the model effectively captured the nuanced features of five distinct breeds. The experimental results highlighted the significant impact of optimizer selection on model performance. Among the optimizers tested, RAdam and SGD exhibited the best performance, achieving an average recall of 96.8%, precision of 97.2%, and mAP50 of 98.1%, making them the most suitable choices for cat breed classification. In contrast, RMSProp showed the lowest performance, particularly in terms of precision and mAP50. These findings underscore the importance of careful optimizer selection in optimizing deep learning models for image classification tasks. Future research could explore several directions to further improve classification accuracy and real-world applicability. First, expanding the dataset to include more cat breeds and incorporating additional visual features, such as coat texture and facial structure, could enhance the model's generalizability. Second, exploring the impact of different data augmentation techniques and hyperparameter tuning may further optimize performance. Additionally, deploying the model on edge devices such as smartphones and smart cameras could enable real-time cat breed identification, making it accessible to pet owners, veterinarians, and conservationists.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA, for funding this research work through project number NBU-FFR-2025-1563-02.

REFERENCES

[1] G. Jocher, J. Qiu, and A. Chaurasia, "Ultralytics YOLO." Jan. 2023, [Online]. Available: https://github.com/ultralytics/ultralytics.

[2] N. Qatrunnada, M. Fachrurrozi, and A. S. Utami, "Cat Breeds Classification Using Convolutional Neural Network For Multi-Object Image," *Sriwijaya Journal of Informatics and Applications*, vol. 3, no. 2, Aug. 2022, https://doi.org/10.36706/sjia.v3i2.46.

[3] K. Jangid, M. Vishwakarma, K. Pal, and L. Rodrigues, "Cat Breed & Emotion Detection Using Yolo, CNN & Canny Edge Detection," *IRE Journals*, vol. 7, no. 8, pp. 1–9, Feb. 2024.

[4] R. C. Chen, V. S. Saravanarajan, and H. T. Hung, "Monitoring the behaviours of pet cat based on YOLO model and raspberry Pi," *International Journal of Applied Science and Engineering*, vol. 18, no. 5, pp. 1–12, 2021, https://doi.org/10.6703/IJASE.202109_18(5).016.

[5] E. Cengil, A. Cinar, and M. Yildirim, "A Case Study: Cat-Dog Face Detector Based on YOLOv5," in *2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, Zallaq, Bahrain, Sep. 2021, pp. 149–153, https://doi.org/10.1109/3ICT53449.2021.9581987.

[6] T. Wang, "Enhanced feline facial recognition: advancing cat face detection with YOLOv8 and TensorRT," in *Fourth International Conference on Computer Vision and Pattern Analysis (ICCPA 2024)*, Sep. 2024, vol. 13256, pp. 193–202, https://doi.org/10.1117/12.3037875.

[7] R. Widyastuti and C.-K. Yang, "Cat's Nose Recognition Using You Only Look Once (Yolo) and Scale-Invariant Feature Transform (SIFT)," in *2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*, Nara, Japan, Oct. 2018, pp. 55–56, https://doi.org/10.1109/GCCE.2018.8574870.

[8] H. T. Hung and R. C. Chen, "Pet cat behavior recognition based on YOLO model," in *2020 International Symposium on Computer, Consumer and Control (IS3C)*, Taichung City, Taiwan, Nov. 2020, pp. 391–394, https://doi.org/10.1109/IS3C50286.2020.00107.

[9] S. Ennaama, H. Silkan, A. Bentajer, and A. Tahiri, "Enhanced Real-Time Object Detection using YOLOv7 and MobileNetv3," *Engineering, Technology & Applied Science Research*, vol. 15, no. 1, pp. 19181–19187, Feb. 2025, https://doi.org/10.48084/etasr.8777.

[10] H. M. Zayani *et al.*, "Deep Learning for Tomato Disease Detection with YOLOv8," *Engineering, Technology & Applied Science Research*, vol. 14, no. 2, pp. 13584–13591, Apr. 2024, https://doi.org/10.48084/etasr.7064.

[11] S. N. Rao, "YOLOv11 Explained: Next-Level Object Detection with Enhanced Speed and Accuracy," *Medium*, Oct. 22, 2024. https://medium.com/@nikhil-rao-20/yolov11-explained-next-level-object-detection-with-enhanced-speed-and-accuracy-2dbe2d376f71.