U.P.B. Sci. Bull., Series C, Vol. 86, Iss. 4, 2024 ISSN 2286-3540

HALYOMORPHA HALYS DETECTION USING NEW YOLO ARCHITECTURES AND NEXT.JS WEB APPLICATION

Marius-Alexandru DINCA[1], Dan POPESCU[2]

Automated detection of insect pests is a critical task to ensure modern agricultural practices, crop health, and productivity. The present study focuses on implementing and optimizing modern YOLO architectures, specifically YOLOv8, YOLOv9, and YOLOv10, for detecting Halyomorpha Halys, an orchard-specific pest. The performance results obtained demonstrate that the YOLOv8 architecture outperforms other modern versions of this family, obtaining higher and balanced metrics and reducing false positive rates. In the same framework, the new models YOLOv9 and YOLOv10 demonstrate competitive performance being viable options for pest detection applications. In parallel, a web application was designed and implemented using the Next.js framework. This application offers a modern, modular, and intuitive interface capable of generating fast predictions based on uploaded images. Combining web techniques with deep learning highlights the potential of these innovative solutions in integrated pest management.

Keywords: orchard monitoring, agriculture, convolutional neural networks, insect detection, web application

1. Introduction

The evolution of agricultural practices has materialized in the field of modern and precision agriculture. Modern agriculture is based on advanced systems to optimize production and sustainability. This field nowadays uses intelligent agricultural machines and advanced IT systems to manage, collect and analyze data in real-time, allowing agricultural staff to make quick and efficient decisions [1, 2]. This data is collected and carefully analyzed to identify the needs of specific agricultural areas, allowing the precise integration of agricultural practices, thereby reducing costs and environmental impact, and increasing agricultural production of various types. Modern and precision agriculture increases productivity and efficiency and contributes to long-term sustainability by ensuring sustainable agricultural practices and responsible use of natural resources and reducing the negative impact on the environment. These features clearly respond to regional and global challenges in terms of food security and climate change. Following this

[1] PhD Student, National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: marius.dinca1411@stud.acs.upb.ro
[2] Prof., National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: dan.popescu@upb.ro

topic, the integration of deep learning in modern agriculture has been an important step for real-time monitoring and analysis of crop images, especially for the control and management of insect pests that can cause major economic losses through crop devastation [3]. Monitoring and controlling insect pests often involve manual, traditional methods. These include classical monitoring by specialized agricultural staff in various forms and pesticide applications, but they are often ineffective, unsustainable, and time-consuming [4]. The technological advance of recent years, especially in the field of deep learning, offers innovative solutions to solve these problems and to characterize modern agriculture through advanced implementations of real-time analysis and monitoring [5]. Looking at this important area, the problem of insect pests is quite pronounced and is found in many crops [6]. Insect pests are a major threat to agriculture affecting crop quality, and their damage results in considerable financial losses for farmers worldwide. Among these challenges the identification and management of pest populations is a critical issue, particularly in orchards where dense and diverse areas can be affected by a multitude of pest species [7]. Within orchards, fruit trees are critical crops and insect pests can cause extensive damage. Examples of such pests include stink bugs, aphids, or codling moths. They can attack leaves and fruits causing deformation, premature fruit loss and, in severe cases, crop destruction [8, 9].

Halyomorpha Halys (HH), popularly known as the brown marmorated stink bug, is a pest of the Pentatomidae family native to East Asia (China, Japan). Due to its massive potential for migration and adaptation, this insect has quickly become one of the major pests in various regions of the world, including Europe and North America [10]. The negative impact on various crops and the economy due to this insect has been the motivation for more research and development focused on studying its behavior, life cycle and control [11]. General appearance characteristics of this insect include an oval-shaped body and a color ranging from dark brown to marble gray. In terms of life cycle, this bug goes through several stages from egg, nymphal stages, and an adult stage. The behavior of this bug is polyphagous, feeding on plant juice by piercing them. HH causes direct damage by feeding on various crops (fruits and vegetables) and causing discoloration, deformation and other types of damage that make agricultural products unmarketable. Its spread affects well-known crops that include fruits (apples, pears), vegetables and some field crops (corn or soybeans) [12]. On the other hand, from an economic and ecological point of view, the losses can be significant [13]. Given these characteristics, deep learning can be used in orchards to monitor the condition of crops and quickly identify the presence of harmful insects, such as *Halyomorpha Halys* [14]. These advanced systems can be integrated to monitor crops in real time and detect signs that could indicate problems or pest infestations, allowing farmers to intervene before damage becomes severe [15, 16]. The present study is aimed at providing such solutions by combining detection models from the YOLO family with a Next.js web application that uses the exported models for the automatic detection of pest insects from digital images. Apart from the introduction, the present work is organized into several sections. Materials and methods section presents the methodology and tools that were the basis of the present study with emphasis on the dataset used, the presentation of the neural networks used, web application and hardware and software characteristics. The next section presents the experimental results and discussions based on the metrics and performances obtained for each detection model. Finally, a conclusion section is introduced to create a summary of the research and outlines the key aspects and future directions of this study.

2. Materials and methods

2.1 Dataset used

The dataset used for training and evaluating the detection models is represented by a series of RGB digital images taken from orchard contexts. The images of the dataset were captured using various devices such as drones, digital cameras, and smartphones. The final dataset contains 500 images illustrating the reference insect HH. The total number of annotated instances was 635. Since the dataset is the basis of a detection model, it was split following a percentage of 70% for training, 20% for validation, and 10% for testing, totaling 350 images for training, 100 for validation, and 50 for testing respectively. Examples of images are presented in Fig. 1.

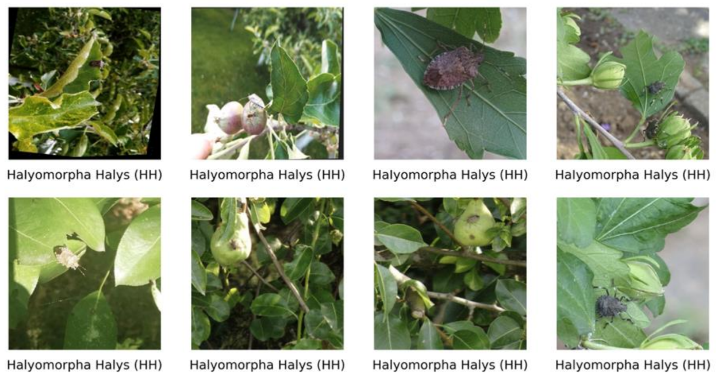

Fig. 1. Example images from the dataset

Image augmentation techniques for the training set were performed to increase the size and diversity of the dataset. Applying various transformations generates new instances that help the detection model to become more robust to variations in the presented data. Some examples in Fig. 1 illustrate augmentation techniques applied to digital images in the dataset. On the other hand, the augmentation process helps to prevent overlearning by exposing the model to a wider range of examples. For the present case, the augmentation part included: flip (vertical and horizontal), rotate (clockwise, counter-clockwise, upside down), shear (+- 10° deg horizontal and vertical), hue (between -15° and +15°), saturation (between -25% and +25%), brightness (between -15% and +15%), exposure (between -10% and +10%), noise (salt and pepper, up to 0.2% of pixels), blur (up to 2px). The input image size for the detection module was 640x640px.

2.2 Neural networks used

The areas represented by orchards represent a complex and variable setting, and the detection of HH pest populations is a critical task considering the power of adaptability and migration of this insect. For the present work, optimized versions of the new state-of-the-art YOLO architectures have been implemented for the automatic detection of HH to demonstrate the performance of this detection model.

The first model chosen for implementation in the present study is YOLOv8, an advanced version of the YOLO family that introduces significant improvements based on the foundations of previous versions [17]. The key features of this new architecture are outlined by changes and innovations on the training and optimization side to successfully manage superior performance in common computer vision tasks, such as object detection. A specialized backbone architecture is seen in this iteration that includes residual convolutional blocks and CSP - Cross-Stage Partial Connections. As with previous versions, the backbone is followed by a Neck architecture featuring PANet (Path Aggregation Network) or FPN (Feature Pyramid Network) blocks. The changes help to efficiently extract features and aggregate them for the detection of objects with various shapes and sizes. Key features are complemented by advanced mosaic and cut mix augmentation mechanisms as well as detection model training optimization tools. The model retains key details for the fast and robust one-stage network characteristic specific to the YOLO family and is built on a framework that allows flexibility, adaptability, and efficiency in real-time applications.

Released in early 2024, YOLOv9 is another advanced iteration of the YOLO family of detection models [18]. The YOLOv9 architecture focuses on solving the problem of loss of information within deep learning architectures during the feature extraction and spatial transformation processes that take place layer by layer. These losses are noted to be the cause of poor model performance and biased gradient flow. Key contributions of YOLOv9 are focused on concepts such as Programmable Gradient Information (PGI) and Generalized Efficient Layer Aggregation Network (GELAN) (Wang, C et al. 2024). Integrating these concepts significantly improves the learning ability of the detection model and ensures that important information is retained throughout the detection and training process. An innovative design for Auxiliary Reversible Branch and Multi-Level Auxiliary Information features components that ensure complete information is preserved and key features are maintained for accurate execution of the detection process. GELAN improves the use of parameters by applying convolution operations and offers flexibility, the architecture being adapted for a wide range of computational blocks, model sizes, and tasks. At the time of study implementation, only YOLOv9c and YOLOv9e versions were available for the YOLOv9 architecture.

YOLOv10 [19] stands out among the latest variants of the YOLO family ensuring a real-time end-to-end object detection architecture. Developed based on the ultralytics architecture, the new architecture design follows the key parts of the YOLO family with significant improvements attached. The backbone architecture introduces an improved version of CSPNet, facilitating feature extraction and improving gradient flow. The neck architecture uses a PAN model for multi-scale feature fusion. The head architecture in this iteration features two key strategies One-to-Many Head and One-to-One Head, responsible for improving the training and prediction generation process, as part of an advanced consistent dual assignment structure. On the other hand, the model capability is extended by using NMS-Free training and a holistic model. They aim to improve performance by using advanced modules such as large-kernel convolutions, lightweight classification heads, and partial self-attention blocks. Finally, this model has the role of generating quality predictions, with low computational cost and adequate efficiency for various scenarios and resources.

2.3 Performance metrics

The evaluation of the detection model was done using common metrics following the object detection task. They were represented by precision, recall, F1 score, mAP@50, and mAP@50:95. Table 1 presents the details of performance indicators and representative calculation formulas. In this sense, the confusion matrix analysis denotes representative indices *TP* – True Positive, *TN* – True Negative, *FP* – False Positive, *FN* – False Negative.

Table 1

Performance indicators for object detection

Indicator	Formula
Precision (*P*)	$P = \frac{TP}{TP + FP}$
Recall (*R*)	$R = \frac{TP}{TP + FN}$
F1 Score (*F*1)	$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$
Mean Average Precision (*mAP*)	$mAP = \frac{1}{N} \cdot \sum_{i=1}^{N} AP_i$

Precision represents the proportion of correct detections out of the total number of detections performed. Recall represents the proportion of correct detections out of the total number of objects that should have been detected. mAP represents a performance indicator specific to the object detection task, which includes an IoU metric (Intersection over Union). IoU measures the overlap between the predicted detection area and the reference area (ground truth). mAP@50 implies the calculation where IoU is 50%. The F1 score is a performance metric that combines precision and recall and is defined as the harmonic mean of precision and recall. Calculating the F1 score is useful when there is an imbalance between classes, providing a balanced performance and being sensitive to both FP and FN types of errors.

2.4 Web application

To complete and validate the present study a web application was created using the JavaScript language and the Next.js framework. This web application can use a detection model exported in ONNX (Open Neural Network Exchange) format and make predictions on the uploaded digital images using the best weights of the trained model. The implemented web application aims to provide a tool to allow users to upload various images and obtain predictions from the detection model exported in the format of their choice. The application architecture is defined in a modular structure involving a backend and a frontend part, for attaching logic and web interface components.

The frontend was implemented using Next.js (React.js). These are popular and established technologies that provide top performance and scalability in web technologies. The web interface is developed by defining and attaching React components with various states and properties and implemented in a secure and robust web application context as part of React and Next.js. The main defined components are represented by a header area that contains the control and navigation part of the web application and a main content area that is responsible for rendering information related to the uploaded images and generating predictions in the web scene. A central web form allows users to select and upload images. The loader component generates an image preview model to display visual information to the user for confirmation. Functionalities of this type are implemented using React Context and State Management tools. The uploaded image starts the inference module, and once the prediction is ready, the resulting information is displayed in the web page component. The results include labels and probabilities associated with the predictions and the resulting image renders this information to display the detected objects. The web interface template, with example predictions, is attached in Fig. 2. As part of the backend area, the web application integrates representative modules for file upload handling and model inference. API Routes using Next.js, and server components are also attached in this context. Next's Route API receives the uploaded image and sends it for processing. The received image is used within the onnx-runtime-web modules to run the exported ONNX model on the image. A Node.js script was developed to process the image and perform the inference. The choice of libraries and frameworks was made based of their state-of-the-art performance, ease of integration, and support for ONNX model deployment. The frontend is developed with Next.js to provide a responsive user interface. The ONNX runtime was implemented for model inference, ensuring compatibility and performance optimization. The design decisions, such as the use of Next.js, React.js and ONNX, were based on their ability to efficiently handle real-time detection and provide a seamless user experience with modern user interface.

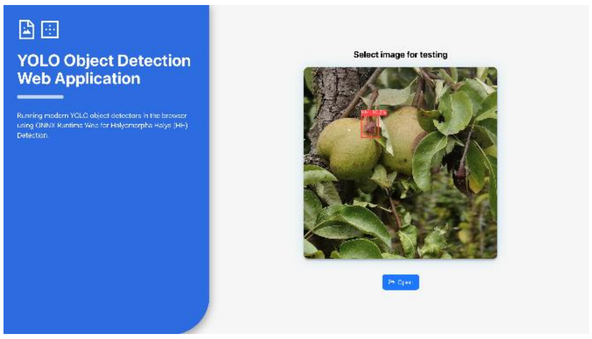
YOLO Object Detection Web Application
Running modern YOLO object detectors in the browser using ONNX Runtime Wasm for Halyomorpha Halys (HH) Detection.
Select image for testing
Open

Fig. 2. Web application interface for HH detection

2.5 Hardware and software used

The development of the detection model of the experiments was done on a custom hardware and software platform. This included an Ubuntu operating system with several important features. The hardware part included an Intel Core i9-11900K CPU, 128GB of RAM, and an ASUS NVIDIA RTX 2080Ti GPU (11GB memory and CUDA support). The implementation details for the hardware and software part significantly influence the computational performance of the chosen architectures. The training was conducted on an NVIDIA RTX GPU which reduced time for training compared to a standard low-end GPU or a CPU. On the other hand, the software stack included Python v3.10, PyTorch and CUDA, optimized for high-performance computing. In this sense the hardware and software resources enhanced the efficiency of both training and inference processes.

3. Experimental Results and Discussions

By integrating a series of key innovations in the neural network architecture, the YOLO architecture proves to be a powerful tool for the applicability of accurate HH detection in orchards. The approach in the present study included transfer learning and fine-tuning techniques to develop the detection model in relation to the previously described insect dataset. Pre-trained weights are used to speed up the process and to optimize the model based on the characteristics of the dataset. The experiments for the detection model included the following parameters for the training and validation part: learning rate 0.01, momentum 0.9, weight decay 0.0005, 300 epochs, batch size 16, workers 8, and SGD optimizer. Table 2 shows the resulting values, calculated based on the expressions of the performance indicators in Table 1. Fig. 3 shows the evolution of the metrics for each detection model.

Table 2

Validation results for YOLO models

Model	**Precision**	**Recall**	***mAP*@50**	***mAP*@50:95**	***F*1 Score**
YOLOv8n	0,981	0,745	0,848	0,504	0.850
YOLOv8s	**0,989**	**0,850**	**0,905**	**0,476**	**0.910**
YOLOv8m	**0,991**	**0,800**	**0,907**	**0,511**	**0.890**
YOLOv8l	0,928	0,800	0,840	0,481	0.860
YOLOv9c	0,889	0,792	0,793	0,386	0.837
YOLOv9e	0,905	0,708	0,815	0,436	0.800
YOLOv10n	0,976	0,788	0,876	0,498	0.870
YOLOv10s	0,988	0,750	0,805	0,452	0.852
YOLOv10m	0,990	0,747	0,815	0,509	0.851
YOLOv10l	0,884	0,762	0,865	0,480	0.820

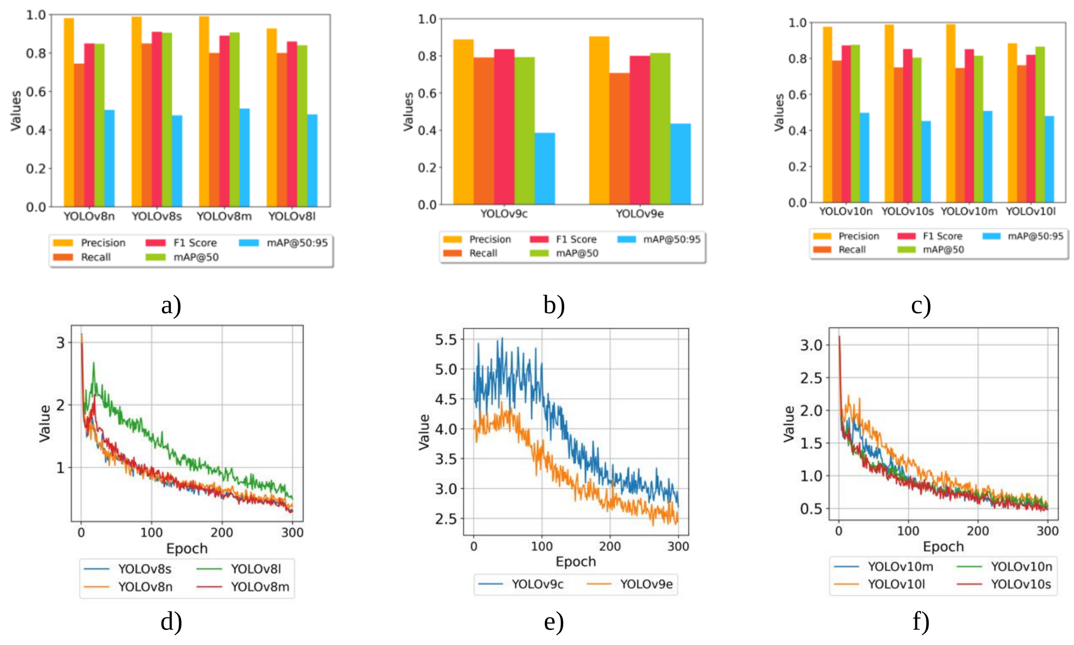

Fig. 3. Validation metrics and evolution of the loss function: a, d. YOLOv8; b, e. YOLOv9; c, f. YOLOv10.

The obtained results demonstrate a good performance in the detection process. The top results have been marked in bold to facilitate comparison with the rest of the implemented models. The study is aimed at comparing the results obtained within each detection model proposed for implementation and optimization. In this step, the characteristics of each architecture are highlighted in the insect identification task.

The analysis of the obtained results shows notable values for the YOLO detection models. The YOLOv8 architecture stands out in this case, providing the best metrics in insect detection. The most balanced results are observed for the YOLOv8s and YOLOv8m variants, combining high precision and good recall. Metrics of this type ensure accurate detection of most insects with a minimal false positive or false negative rate. The YOLOv10 detection model shows solid performance, but generally below YOLOv8. In this case, the YOLOv10n, s, and m variants are representative, providing metrics relevant to this study. Although competitive the models may have slight insect misidentification problems with poor detection values compared to YOLOv8. Lower performance is observed for the YOLOv9 model compared to YOLOv8 or YOLOv10, indicating a higher risk of missing insect pests to be detected. The moderate performances of this YOLOv9 architecture indicate the need for careful optimization and improvements on the part of the architecture and algorithms, capable of providing solutions to these limitations for the detection of insects in the presented scenarios. For the detection of HH insects at the orchard level, the performances of the detection models suggest that such architectures are suitable to be developed and implemented in practice in agriculture, minimizing losses associated with insect pest populations and ensuring effective protection.

The technical implementation demonstrates the ability to combine modern web technologies and deep learning to create an effective prediction solution in the web browser. Uploading images start at the interface accessed by the user using accordingly validated web forms that send the digital image to a specific endpoint. For the object detection experiments, using the implemented web application, the image processing is required within this API endpoint to match model features (resizing, normalizing, tensors). Using the inference model, the processed image with the relevant results is sent by the API back to the frontend where they are rendered within the interface and prediction components. Fig. 4 shows examples of images with predictions resulting from the testing process.

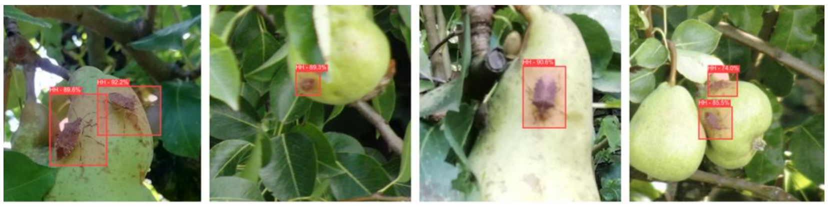

Fig. 4. Example images with predictions

The methodology for the study is characterized by important steps that start from the data acquisition step, up to the end-user application. The first step is to collect images of HH using high-resolution cameras. The data is then preprocessed to enhance image quality and annotated for training purposes. Subsequently, the annotated images are used to train various YOLO neural network architectures. The trained models are evaluated using standardized metrics, and the best-performing model is deployed in a web application for real-time detection.

The approach described above has numerous advantages. These are performance and scalability, portability, availability, superior user experience, as well as security and control over data flow. The scientific contribution is noted by addressing the gap between advanced neural network models and their practical application in agriculture, demonstrating the potential of intelligent systems in mitigating economic damage in orchards. The web application enhances the usability and accessibility of the detection system. It serves as an interface between the trained models and the end-users, typically farmers or agricultural experts. By providing real-time detection capabilities, the application enables prompt identification and mitigation of HH infestations.

Analyzing and testing the architectures of the present study, top performances are observed for most of the models and variants chosen for implementation. However, maximum detection and accurate identification performance of the reference insect is noted when it is visible under favorable conditions. However, various observations regarding the evolution of metrics and the general detection can be noted. The main problem in identifying this type of insect is its presence in the context of orchards and trees. Most of the time the insect is partially hidden, among leaves and tree crowns, and this context has a considerable negative impact on the detection process. Other limitations observed are represented by the fact that the insect can be easily confused with some artifacts visible within the crops, the color of the insect being close to that of the branches, the small textures of the trees, or the various spots on the leaves. Another problem observed that influences the detection process is the blur effect that can affect the identification of the insect, with the models having trouble extracting the features necessary for correct identification, losing key information such as the shape, texture, and specific details of the insect.

The limitations observed throughout the study open the way to new directions of development and research to solve the problems that have arisen and to identify the pests under various unfavorable conditions. The first step in this direction of future research is the increase of the data set, which will favor this study, in the correct identification of the insect regardless of its size. A second step can integrate new classes into the dataset, often pest populations have various stages of emergence and development, especially for HH (nymphs, adults, etc.). Other species can be attached in the same context. Careful optimization of the proposed architectures may include new feature extraction methods, particularly targeting specific backbone, neck, or head architectural areas.

4. Conclusions

The study of YOLO architectures for HH detection of using digital imagery taken from the orchard level revealed superior performance. In this study, variants of the YOLOv8, YOLOv19, and YOLOv10 models were studied and optimized for accurate detection of the reference insect. A specific and comprehensive dataset was used to train and evaluate the YOLO detection models. The proposed analysis and evaluation of the detection models revealed the superiority of the YOLOv8 architecture for these tasks, especially YOLOv8s and YOLOv8m, balancing precision and recall metrics, providing satisfactory global accuracy, and minimizing false positives and negatives. In contrast, the YOLOv9 and YOLOv10 models achieved lower performance and failed to match the performance achieved by YOLOv8. Now, the YOLOv8 architecture can be highlighted as the most suitable for practical implementation, in solutions with automated systems, providing accurate and consistent detection of HH, and ensuring effective crop protection. However, the capabilities and key features of the YOLOv9 and YOLOv10 models can be noted in this framework to extend the study by generating detection architectures that can outperform existing state-of-the-art models, leveraging the associated flexibility and performance. In addition to the study, a Next.js web application is presented that integrates web and deep learning technologies to make predictions on uploaded images, using detection models exported in a common ONNX format. This implementation optimizes inference and allows the application to run on different platforms and in relation with different services. The user experience is improved by offering a modern and interactive solution. The presented solution is robust and performant and manages to ensure the safe functionalities in fast prediction based on digital images.

R E F E R E N C E S

[1]. L. Butera, A. Ferrante, M. Jermini, M. Prevostini, and C. Alippi, “Precise Agriculture: Effective Deep Learning Strategies to Detect Pest Insects,” IEEE/CAA Journal of Automatica Sinica, no. 2, pp. 246–258, Feb. 2022, doi: 10.1109/jas.2021.1004317.

[2]. D. Popescu, A. Dinca, L. Ichim, and N. Angelescu, “New trends in detection of harmful insects and pests in modern agriculture using artificial neural networks. a review,” Frontiers in Plant Science, Nov. 2023, doi: 10.3389/fpls.2023.1268167.

[3]. Hu, X. Deng, Y. Lan, X. Chen, Y. Long, and C. Liu, “Detection of Rice Pests Based on Self-Attention Mechanism and Multi-Scale Feature Fusion,” Insects, no. 3, p. 280, Mar. 2023, doi: 10.3390/insects14030280.

[4]. M. Kereselidze, D. Pilarska, N. Guntadze, and A. Linde, “Halyomorpha halys Stål, (Hemiptera:Pentatomidae) feeding effects on some agriculturalfruits in Georgia,” Turkish Journal of Zoology, no. 3, pp. 298–303, Jan. 2022, doi: 10.55730/1300-0179.3058.

[5]. A. Kargar, M. P. Wilk, D. Zorbas, M. T. Gaffney, and B. Q’Flynn, “A Novel Resource-Constrained Insect Monitoring System based on Machine Vision with Edge AI,” 2022 IEEE 5th International Conference on Image Processing Applications and Systems (IPAS), Dec. 2022, doi: 10.1109/ipas55744.2022.10052895.

[6]. Coulibaly, B. Kamsu-Foguem, D. Kamissoko, and D. Traore, “Explainable deep convolutional neural networks for insect pest recognition,” Journal of Cleaner Production, p. 133638, Oct. 2022, doi: 10.1016/j.jclepro.2022.133638.

[7]. D. Popescu, L. Ichim, M. Dimoiu, and R. Trufelea, “Comparative Study of Neural Networks Used in Halyomorpha Halys Detection,” 2022 30th Mediterranean Conference on Control and Automation (MED), Jun. 2022, doi: 10.1109/med54222.2022.9837254.

[8]. J. Liu and X. Wang, “Tomato Diseases and Pests Detection Based on Improved Yolo V3 Convolutional Neural Network,” Frontiers in Plant Science, Jun. 2020, doi: 10.3389/fpls.2020.00898.

[9]. L. Liu et al., “PestNet: An End-to-End Deep Learning Approach for Large-Scale Multi-Class Pest Detection and Classification,” IEEE Access, pp. 45301–45312, 2019, doi: 10.1109/access.2019.2909522.

[10]. G. Noël, A. Segers, J. Carpentier, L. Rossini, E. Garone, and F. Francis, “An update for Halyomorpha halys (Stål, 1855) (Hemiptera, Pentatomidae) distribution in Belgium,” Biodiversity Data Journal, Jun. 2024, doi: 10.3897/bdj.12.e125067.

[11]. G. Vétek, B. Károlyi, Á. Mészáros, D. Horváth, and Korányi, “The invasive brown marmorated stink bug (Halyomorpha halys) is now widespread in Hungary,” Entomologia Generalis, no. 1, pp. 3–14, Oct. 2018, doi: 10.1127/entomologia/2018/0631.

[12]. F. Sanna et al., “Halyomorpha halys (Hemiptera: Pentatomidae) as the major contributor to early olive drop in northern Italy,” Journal of Economic Entomology, Jun. 2024, doi: 10.1093/jee/toae126.

[13]. A. Regmi, K. Neupane, and J. Harper, “Grower assessment of profitability impact from Brown Marmorated Stink Bug, Halyomorpha halys (Stål), the value of management information sources, and use of potential management practices,” Journal of Applied Entomology, May 2024, doi: 10.1111/jen.13277.

[14]. A. Dinca, D. Popescu, C. M. Pinotti, L. Ichim, L. Palazzetti, and N. Angelescu, “Halyomorpha Halys Detection in Orchard from UAV Images Using Convolutional Neural Networks,” in Advances in Computational Intelligence, Springer Nature Switzerland, 2023, pp. 315–326.

[15]. M. Dai, M. M. H. Dorjoy, H. Miao, and S. Zhang, “A New Pest Detection Method Based on Improved YOLOv5m,” Insects, no. 1, p. 54, Jan. 2023, doi: 10.3390/insects14010054.

[16]. A. Dinca, N. Angelescu, L. Ichim, and D. Popescu, “Deep Neural Networks for Halyomorpha Halys Detection,” 2023 31st Mediterranean Conference on Control and Automation (MED), Jun. 2023, doi: 10.1109/med59994.2023.10185751.

[17]. Jocher, G., Chaurasia, A., & Qiu, J. (2023). Ultralytics YOLO (Version 8.0.0) [Computer software]. https://github.com/ultralytics/ultralytics

[18]. C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, “YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information,” arXiv, 2024. [Online]. Available: https://arxiv.org/abs/2402.13616

[19]. A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, “YOLOv10: Real-Time End-to-End Object Detection,” arXiv, 2024. [Online]. Available: https://arxiv.org/abs/2405.14458