

Deep Learning-Based Pest Insect Detection for Autonomous Agricultural Systems: Development and Field-Testing of a Custom Convolutional Neural Network Architecture

Selim Sürücü¹, Berk Küçük², Mahammad Karimzade³

¹Department of Computer Engineering, Çankırı Karatekin University, Çankırı, 18100, Türkiye, Orcid: 0000-0002-8754-3846

²Department of Computer Engineering, Çankırı Karatekin University, Çankırı, 18100, Türkiye, Orcid: 0009-0005-3500-4540

³Department of Electrical Engineering, Çankırı Karatekin University, Çankırı, 18100, Türkiye, Orcid: 0009-0002-1830-3758

Article Info

Article history:

Received Dec 23, 2025

Revised Apr 19, 2026

Accepted Jun 22, 2026

Keywords:

Insect Classification

Precision Agriculture

Autonomous Systems

Convolutional Neural Networks

Edge Computing

ABSTRACT

Global food security is under increasing threat from pest organisms that target agricultural production. The broad-spectrum pesticides traditionally used to combat these pests cause permanent damage to ecosystems, pose risks to human health, and lead to a critical decline in biodiversity. Precision agriculture technologies, as part of the Agriculture 4.0 revolution, offer innovative solutions to these problems. This study focuses on developing a novel artificial intelligence model capable of detecting and classifying pest insects with high accuracy, to serve as the intelligence for autonomous spraying systems. Within the scope of this research, a comprehensive and diverse dataset of over 12,000 images was meticulously compiled, featuring a total of 11 insect species, including 8 economically significant pests and 3 beneficial species for the ecosystem. To enhance the robustness and diversity of the dataset, advanced Generative Adversarial Networks (GANs) such as StyleGAN2-ADA were used alongside traditional data augmentation techniques to generate synthetic images. On this rich dataset, VGG-16 and ResNet152 V2 models, representing the transfer learning approach, were developed and comparatively analyzed against a lightweight Convolutional Neural Network (CNN) architecture custom-designed for resource-constrained edge computing devices. While the ResNet152 V2 model achieved the highest accuracy (92.4%) in laboratory tests, the custom-designed CNN model proved to be the most practical and effective solution in real-world field tests conducted with a Raspberry Pi-based prototype, delivering 82.5% accuracy, a much smaller model size (31 MB), and superior processing speed (7.5 FPS). These results strongly demonstrate that application-specific, lightweight models can exhibit more robust and efficient performance in real-world scenarios compared to complex and large transfer learning architectures. The developed system has the potential to make significant contributions to agricultural sustainability and food security by optimizing pesticide use.

Copyright © 2026 Institute of Advanced Engineering and Science.
All rights reserved.

Corresponding Author:

Selim SÜRÜCÜ

Department of Computer Engineering, Çankırı Karatekin University

Çankırı, 18100, Türkiye,

Email: selimsurucu@karatekin.edu.tr

1. INTRODUCTION

The responsibility of feeding our planet's growing population places unprecedented pressure on the agricultural sector. This pressure is further exacerbated by factors such as global climate change, water scarcity,

and the reduction of arable land, alongside biological threats that endanger food security. The Food and Agriculture Organization of the United Nations (FAO) reports that up to 40% of global agricultural output is lost due to various pests, primarily pest insects [1]. The main strategy employed for decades to combat these massive losses has been the widespread and often indiscriminate application of pesticides. However, the cost of this "one-size-fits-all" approach has been severe: chemical contamination of soil and water resources, the emergence of pesticide-resistant super-pests, mass deaths of beneficial organisms crucial to ecosystems such as bees and ladybugs, and the accumulation of toxic residues in the human food chain [2].

The way to break this unsustainable cycle is through a paradigm shift in agricultural practices, through the fourth agricultural revolution known as "Agriculture 4.0". This revolution aims to make agriculture smarter, more efficient, and more environmentally friendly by using technologies such as artificial intelligence (AI), the Internet of Things (IoT), robotics, drones, and big data analytics. At the heart of this vision lies the concept of "precision agriculture." Precision agriculture aims to maximize efficiency and minimize environmental impact by analyzing the needs of each plant or small area in a field individually and applying resources (water, fertilizer, pesticides) only where, when, and in the amount needed [3]. One of the most powerful applications of this approach is autonomous spraying systems—intelligent machines that scan the field to detect pests and apply treatment only to that specific spot. The success of these systems depends on a fundamental capability: a "vision" system that can instantly and accurately distinguish between harmful and harmless organisms.

The existing literature in this field presents promising studies on deep learning and image processing-based insect detection systems [4, 5]. However, a significant portion of these studies has various limitations. They often use datasets consisting of images taken under controlled laboratory conditions with a limited number of species, a homogeneous background, and ideal lighting. This leaves unanswered the question of how robust the developed models are against the complexities of real-world conditions (e.g., variable light, wind, insects hiding behind leaves, motion blur). Furthermore, high-performance deep learning architectures like VGG and ResNet have millions of parameters and require significant computational power. This makes it practically impossible to run them in real-time on edge computing devices such as autonomous drones or tractors, which have limited battery and processing power [6, 7].

Despite the rapid advancements in deep learning for pest detection, there remains a significant gap between success in laboratory settings and field applications. Most of the current high-performance models require a significant amount of computational resources (for example, high-end GPUs). This makes them incompatible with the limited battery (power) capacity of autonomous agricultural drones or small-scale robotic sprayers. Additionally, in studies conducted with various datasets due to environmental factors in real-world agriculture, such as variable lighting and noise during movement, classification success is seen to drop significantly. Models in the literature experience a significant drop in reliability when they are taken out of controlled environments. This study addresses these limitations by proposing a vertically integrated approach. This study utilizes GANs to simulate field conditions during training and optimizes a specifically lightweight CNN for edge inference on low-cost hardware. The innovative aspect of this study lies not in a non-standard machine learning workflow, but in the conscious vertical integration of data-generating synthesis (StyleGAN2-ADA) with parameter-optimized sequential design.

2. LITERATURE REVIEW

Nyakuri et al. developed a lightweight CNN model called Tiny-LiteNet for the rapid and accurate detection of agricultural pests on edge devices. The model is only 1.2 MB in size and contains approximately 1.48 million parameters. It is designed to operate independently of the cloud on a Raspberry Pi 5 with a GSM/GPRS module. When compared to common deep learning architectures such as VGG16, Inception3, ResNet50, DenseNet121, EfficientNetB0, and MobileNetV2, Tiny-LiteNet stands out with both its smaller model size and significantly higher efficiency. In laboratory tests, it achieved 98.6% accuracy, demonstrating that high-precision insect detection is possible even on small, low-power hardware [8]. In another study, higher accuracy and speed were achieved in the detection of insect and plant diseases by adding the Global Attention Module (GAM) and DGSCConv layers to the YOLOv7-Tiny architecture. Additionally, the model's training stability was enhanced with SiLU activation and the SIOU loss function. Experiments conducted on the tomato leaf dataset achieved 95.53% precision. The model, which has 4.43 million parameters and a computational complexity of 10.2 GFLOPs, demonstrated its suitability for real-time agricultural applications by offering an extraordinary speed of 168 FPS [9].

In another study, Huang et al. proposed a lightweight and optimized model called YOLO-YSTs based on YOLOv10n for detecting small and dense insects in sticky traps. The model uses SPD-Conv modules, iRMB residual attention structures, and Inner-SIoU loss function, which have been shown to increase the success rate in detecting small targets. According to experimental results, the model achieved 83.2% precision. Additionally, it was observed to have a high speed of 139 FPS and a computational complexity of 8.8 GFLOPs [10]. In another study that improved the performance of insect detection systems using GAN-based data augmentation methods in scenarios where whitefly images were limited, the dataset, which initially contained only 140 images, was increased fourfold through GAN-based synthetic data generation. In experiments conducted with the YOLO architecture, the recall value increased from 54.4% to 93.2%, and the average IoU increased from 34.6% to 70.9%. GAN data provided an additional 1.4% increase in recall and 2.6% increase in IoU compared to classical augmentation methods. Human subjects reported that GAN-generated data could be distinguished from real images at a rate of 56–67%. This result demonstrates that GAN-based augmentation can significantly improve the accuracy of insect detection models under low-data conditions [11].

3. MATERIALS AND METHODS

Our research methodology is built on a repeatable and scientifically sound foundation, consisting of four main stages: (1) Multi-source data collection and creation of a comprehensive insect image dataset, (2) Advanced data preprocessing and enrichment with synthetic data generation, (3) Design and training of deep learning models representing different approaches, and (4) Systematic testing and evaluation of the models on a Raspberry Pi-based prototype under real field conditions.

3.1. Dataset Development

The performance of the deep learning model is directly related to the quality and diversity of the data on which the model is trained. In line with this principle, a dataset covering a total of 11 insect species has been created, including 8 harmful species that cause economic losses in agricultural production and 3 beneficial/harmless species that play a key role in ecosystem balance. While determining these insect species, those commonly found in Turkey's agricultural areas were selected. By focusing on these regional species, a functional solution is offered for local autonomous systems. A multi-channel strategy was followed in the data collection process:

- **Open-Access Databases:** Publicly available platforms such as Kaggle, GitHub, ImageNet, and various university botanical archives were scanned to create a foundational image pool.
- **Academic Literature Review:** High-resolution insect images published and accessible in reputable journals in the fields of entomology and agricultural sciences were compiled.
- **Original Field Studies:** This stage, which constitutes the most valuable data source of the project, involved fieldwork in wheat fields, orchards, and vegetable greenhouses on and around the Çankırı Karatekin University campus. Images were captured in different seasons, at different times of the day (07:00-10:00, 12:00-15:00, 17:00-20:00), and under various weather conditions (sunny, cloudy, windy) to capture environmental variability.

The collected 8050 original images were meticulously labeled and verified by an entomology expert. Details regarding the species in the dataset and their agricultural importance are presented in Table 1. The distribution of pest and beneficial species in the dataset is detailed in Figure 1.

3.2. Data Preprocessing and Augmentation

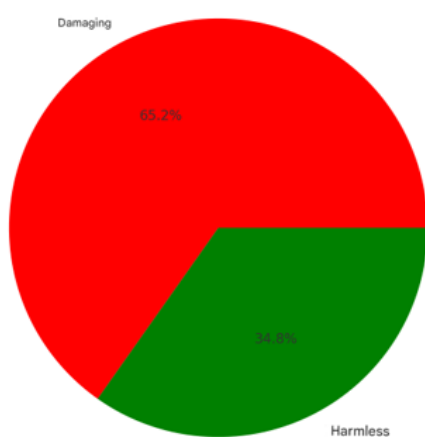
The raw data underwent a comprehensive preprocessing and augmentation process to be converted into a format that deep learning models can "understand" and to maximize the model's learning capacity. The steps of this process are illustrated in Figure 2.

- **Data Cleaning and Standardization:** Low-resolution, out-of-focus, incorrectly labeled, or images where the insect was too small to be recognized were manually filtered out. The remaining 7,850 high-quality images were resized to the commonly used 224x224 pixel dimension to be compatible with the input layer of all models. Pixel values were normalized from the [0, 255] range to the [0, 1] range to ensure a more stable training process.

Table 1. Insect species, classes, and image counts in the dataset

Class	Species (Scientific Name)	Agricultural Impact / Role	Image Count (Original)
Pest	Aphid (Aphidoidea)	Sucks plant sap, spreads viruses	735
Pest	Spider Mite (T. urticae)	Causes yellowing and drying of leaves	684
Pest	Whitefly (Aleyrodidae)	Sucks plant sap, creates sooty mold	612
Pest	Thrips (Thysanoptera)	Causes deformation in leaves and flowers	588
Pest	Armyworm (Spodoptera)	Feeds by eating leaves and stems	654
Pest	Codling Moth (C. pomonella)	Damages fruit by entering it	625
Pest	Colorado Potato Beetle (L. decemlineata)	Damages potato and other Solanaceae plants	702
Pest	Wireworm (Agriotes)	Damages roots and tubers	650
Harmless	Ladybug (Coccinellidae)	Predator that eats pests like aphids	951
Harmless	Bee (Anthophila)	Vital for pollination	982
Harmless	Butterfly (Lepidoptera)	Assists in pollination	867
Total	11 Species	-	8,050

Harmful and Harmless Insect Image Distribution



Distribution of Images According to Insect Species

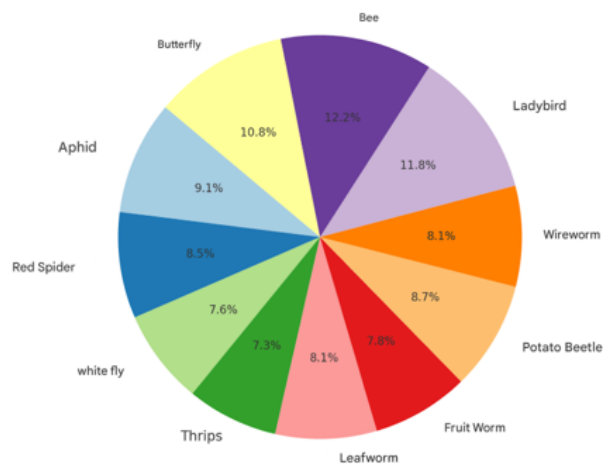


Figure 1. Pest and Harmless Insect Image Distribution

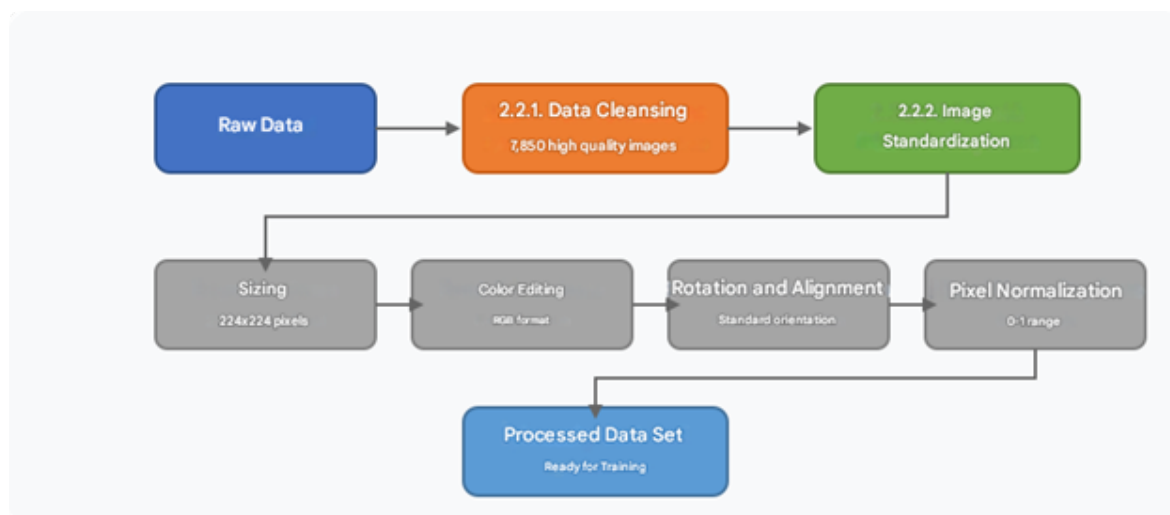


Figure 2. A flowchart illustrating the image preprocessing pipeline

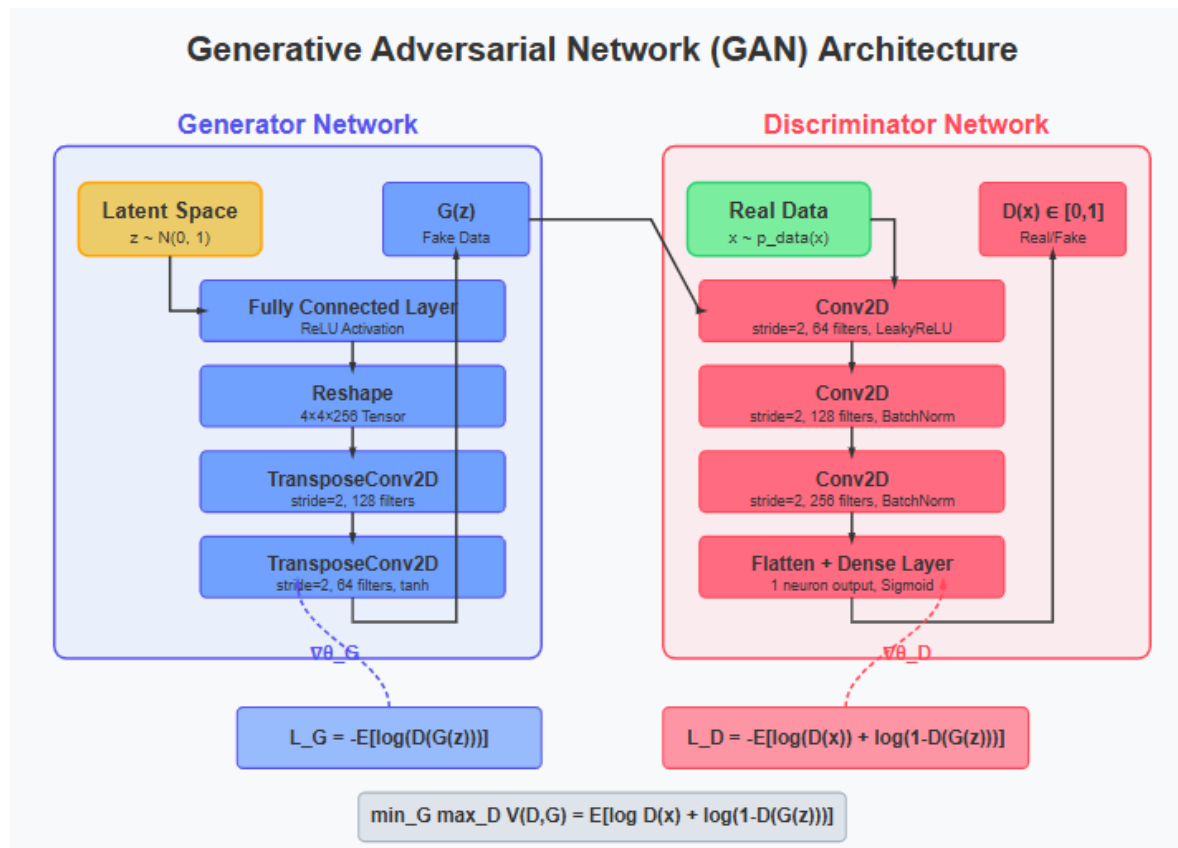


Figure 3. Generative Adversarial Network (GAN) Architecture

- **Data Augmentation:** To increase the model's generalization ability by learning different variations of the same insect and to prevent overfitting, a two-tiered augmentation strategy was applied:
 - **Traditional Transformations:** Geometric and photometric transformations such as horizontal/vertical flipping, random rotation within ± 15 degrees, random zooming/cropping by 10-20%, and slight adjustments to brightness and contrast values were applied to each image in the training set on-the-fly during training.
 - **Generative Adversarial Networks (GANs):** The StyleGAN2-ADA (Adaptive Discriminator Augmentation) architecture was chosen to add completely new and realistic examples to the dataset, particularly due to its superior performance in training with limited data [12]. A separate GAN model was trained for each insect species using the original images of that species. After training was complete, 500 high-quality and diverse synthetic images were generated from each model. The fundamental architecture of the Generative Adversarial Network used is shown in Figure 3. The StyleGAN2-ADA architecture was used not only for data augmentation but also to introduce realistic environmental noise into the training cycle. Environmental effects such as low-frequency motion blur, thanks to the generative model, also enabled the specialized CNN to produce images capable of handling dynamic lighting and physical vibrations encountered during autonomous field operations.

These 4,500 synthetic images, after being quality-checked by an expert, were added only to the training set. As a result of these processes, the total dataset reached 12,350 images (7,850 original + 4,500 synthetic). The dataset was split into three parts using stratified sampling to maintain the class ratios: 70% for training (8645 images), 15% for validation (1853 images), and 15% for testing (1852 images).

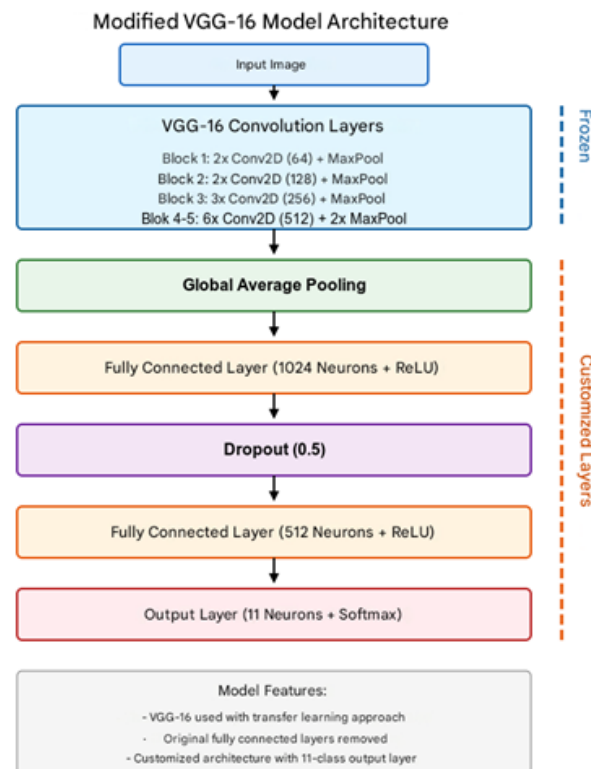


Figure 4. Modified VGG-16 Model Architecture

3.3. Model architectures

In this study, three different deep learning architectures were designed, trained, and compared to solve the problem. This selection represents the two main approaches in the literature (transfer learning and design from scratch).

3.3.1. VGG-16 based model

The VGG-16 architecture [13], known for its simplicity and effectiveness, was used as a basic transfer learning model. Its architecture is detailed in Figure 4. The weights of the convolutional layers pre-trained on ImageNet were frozen and used as feature extractors. On top of this frozen base, a Global Average Pooling (GAP) layer, followed by fully connected layers and finally an 11-class Softmax output layer were added.

3.3.2. ResNet152 V2 based model

ResNet152 V2 [14], which solves the vanishing gradient problem in very deep networks using residual blocks, was used as a base. Its modified architecture is shown in Figure 5. In this model, a more flexible “fine-tuning” strategy was followed. Initially, all pre-trained layers were frozen, and only the newly added classifier head was trained. In the later stages of training, the learning rate was reduced, and the weights of the top residual blocks were also unfrozen and included in the training process.

3.3.3. Custom convolutional neural network (CNN)

This model, which constitutes the main innovation of this study, was designed from scratch to operate with high efficiency, especially on resource-constrained edge computing devices, as shown in Figure 6. The design philosophy of the architecture is to strike a balance between effectiveness and lightness. The model consists of five convolutional blocks with increasing filter counts ($32 \rightarrow 512$). Each block contains Batch Normalization, ReLU activation, and Max Pooling layers. At the end, a Global Average Pooling (GAP) layer is used to reduce the number of parameters, and a Dropout layer is included to prevent overfitting.

The design of this special CNN architecture has been kept quite simple. The reason for its simplicity is the adoption of an optimization strategy with hardware awareness specifically tailored for agricultural edge

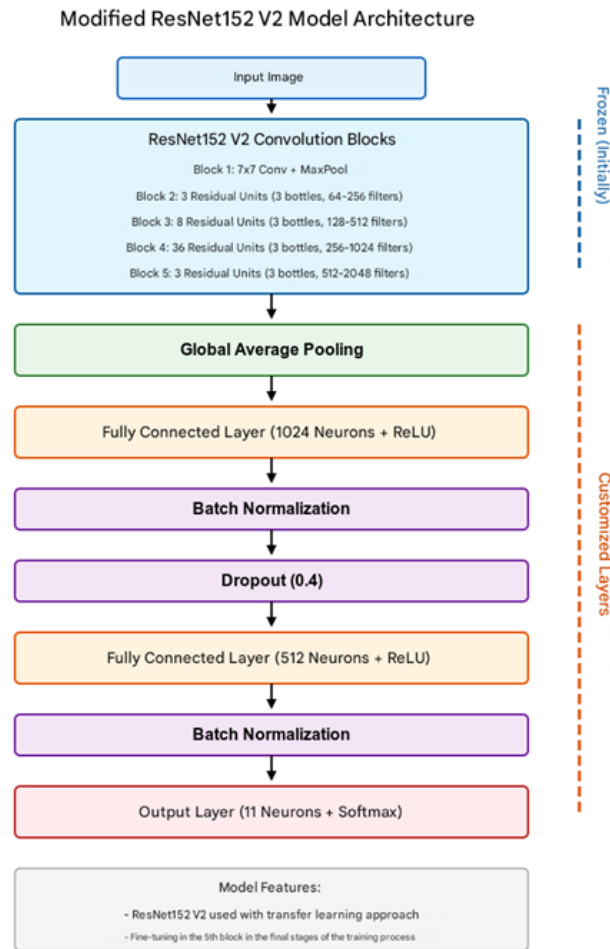


Figure 5. Modified ResNet152 V2 Model Architecture

computing. Instead of the computational cost of general-purpose deep architectures, this architecture aims to minimize floating-point operations (FLOPs) to achieve high inference speeds on the Raspberry Pi 4B platform. The decision to use five consecutive convolutional blocks was made to address the need for low memory usage (31 MB) and to reduce power consumption during prolonged field operations, instead of using complex connections or attention mechanisms. This 'lightweight' approach has allowed the model to strike a balance between satisfactory field accuracy (82.5%) and practical applicability (7.5 FPS). This makes it more suitable for low-cost autonomous systems compared to parameter-weighted alternatives.

3.4. Model training and real-time test system

All models were trained on a single NVIDIA Tesla V100 GPU to accelerate model training, using the Pytorch deep learning library. Standard hyperparameters such as the Adam optimizer, Categorical Cross-Entropy loss function, and a batch size of 32 were used for the training process. An Early Stopping mechanism was activated to prevent overfitting. The prototype system set up for real-world tests consists of cost-effective and widely available components:

- **Hardware:** A Raspberry Pi 4B (4GB RAM) was used as the brain of the system, and a high-resolution Raspberry Pi Camera Module V2 (8MP Sony IMX219 sensor) as its eye. The main reason for preferring high-end platforms like the NVIDIA Jetson Orin Nano or hardware-accelerated Raspberry Pi 5 is that, although these devices offer superior operational efficiency, their high costs make them inaccessible technologies for local farmers in regions like Turkey. A 10,000 mAh portable power supply was integrated for the system to operate autonomously in the field.

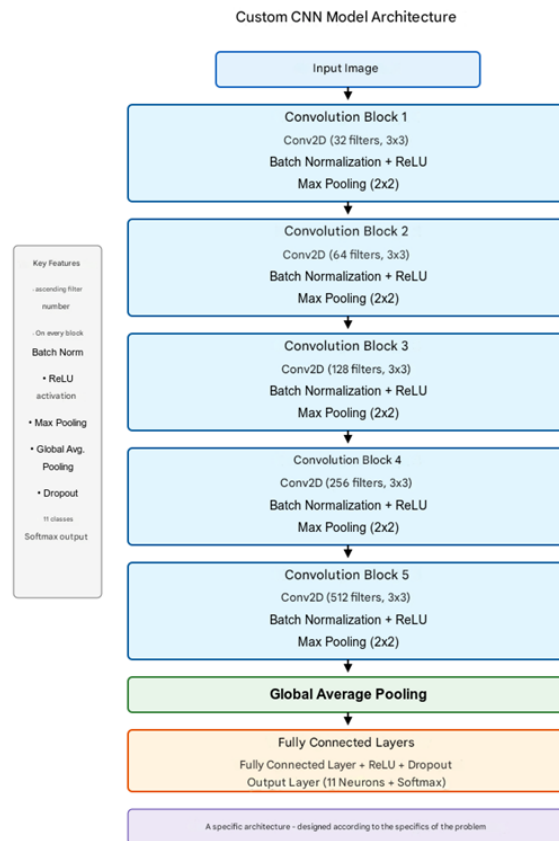


Figure 6. Custom CNN Model Architecture

- **Software Optimization:** To ensure the models run efficiently on a device with limited processing power like the Raspberry Pi, the models trained in Pytorch were optimized using NVIDIA's TensorRT library. This process significantly increases inference speed.

4. RESULTS AND DISCUSSION

4.1. Laboratory environment performance comparison

The quantitative performance of the three trained models was comprehensively evaluated on the test dataset (1,852 images), which they had never seen before. Standard metrics such as Accuracy, Precision, Recall, and the F1-Score were used for evaluation. The training progress of the models is documented in Figure 7. The final quantitative results are summarized in Table 2 and visualized in Figure 8.

Table 2. Comprehensive Performance Comparison of Models on the Test Dataset

Model	Accuracy (%)	Precision	Recall	F1-Score	Model Size (MB)	Inference Time (ms/image)
VGG-16	88.7	0.889	0.887	0.885	528	105
ResNet152 V2	92.4	0.925	0.924	0.923	232	142
Custom CNN	90.9	0.912	0.909	0.910	31	87

The results of the laboratory tests shed light on several important points. As expected, ResNet152 V2, with its extremely deep 152-layer architecture and residual connections, showed the highest classification performance with 92.4% accuracy. However, this performance comes at a cost: a model size of 232 MB and the slowest processing time. On the other hand, the Custom CNN model, the main contribution of our study, performed very close to ResNet with 90.9% accuracy, while establishing overwhelming superiority in practicality metrics (31 MB model size, 87 ms processing time).

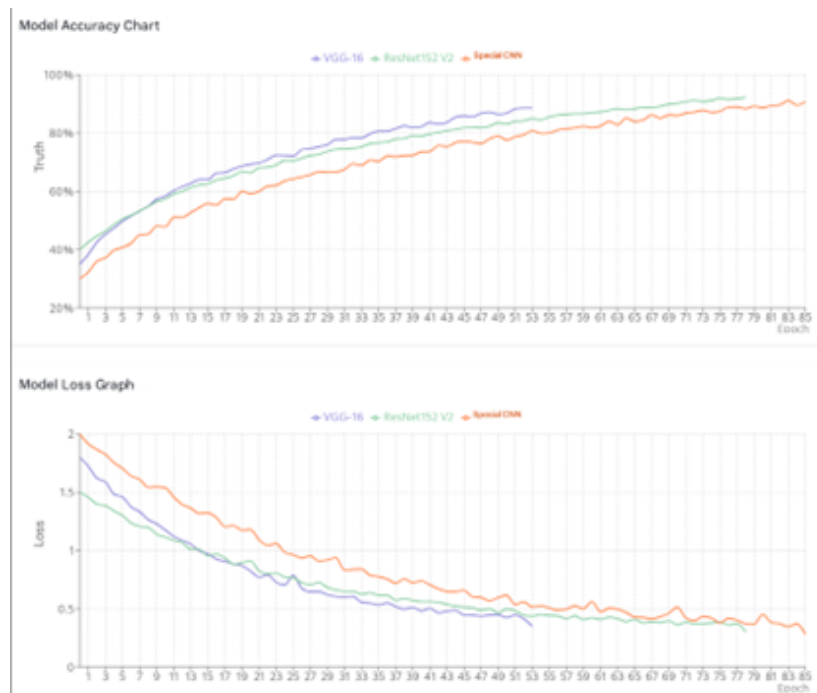


Figure 7. Model Accuracy Graph

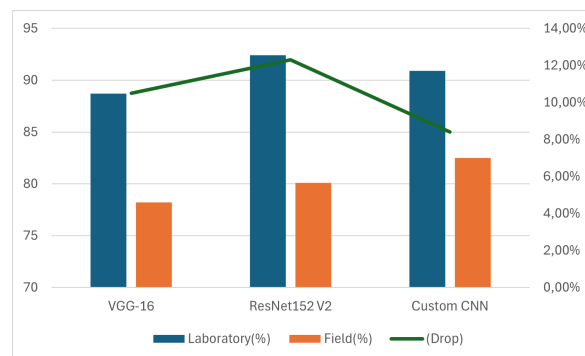


Figure 8. Result of Truth

4.2. Real-World field test results

A model's success in the lab does not guarantee its value in the real world. Therefore, to measure the practical applicability of the models, we conducted extensive field tests in agricultural lands in Çankırı with our prototype system. The tests were carried out at different times of the day, under different weather conditions, and on different vegetation. The results of the inferences made on more than 500 real-world images are presented in the comparative graph in Figure 8. The real-world tests yielded extremely illuminating and important results:

- **Inevitable Performance Drop:** As expected, the performance of all models experienced a drop in the range of 8-12% in real field conditions compared to the laboratory environment. This drop is due to uncontrollable factors (variable light, motion blur, partial occlusion, complex background).
- **Robustness and Superiority of the Custom CNN Model:** The most striking finding was that the Custom CNN model, which ranked second in the lab, showed the best performance in the real world with an accuracy rate of 82.5%. The performance of ResNet152 V2 fell behind that of the Custom CNN. This indicates that the simpler, custom-designed CNN architecture is more robust against the "noisy" and imperfect data of the real world. Field tests conducted on land belonging to Çankırı Karatekin University

Table 3. Comparison with studies in the literature

Paper	Model	Year	Accuracy / mAP	FPS	Model Size / Parameters
[8]	Tiny-LiteNet (Raspberry Pi 5)	2025	98.6%	-	1.2 MB (~1.48 M param.)
[9]	DGS-YOLOv7-Tiny	2025	Precision: 95.5%, Recall: 92.9%, mAP@0.5: 96.4%	168 FPS	~4.43 M param., 10.2 GFLOPs
[10]	YOLO-YSTs (Sticky Trap)	2025	Precision/Recall: 83.2%, mAP@50: 86.8%, mAP@50-95: 41.3%	139 FPS	8.8 GFLOPs
[11]	GAN-Augmented YOLO (Whitefly)	2022	Recall: 93.2%, Avg IoU: 70.9%	-	-
Our paper	Raspberry Pi prototype (Custom CNN+ VGG-16 + ResNet152V2 comparison)	2025	ResNet152V2: 92.4% (lab) Custom CNN: 82.5% (field)	Custom CNN: 7.5 FPS	31 MB (Custom CNN)

were carried out on rugged terrain. The resulting 8.4% drop in accuracy (from 90.9% in the laboratory to 82.5% in the field) is believed to stem from various environmental factors, such as real-time motion blur and changes in light intensity. The optimized 5-block architecture of the specialized CNN ensured that the impact of this local pixel noise remained limited.

- **Real-Time Applicability:** In terms of practicality, the superiority of the Custom CNN model became even more apparent. After optimization with TensorRT, it achieved an average processing speed of 7.5 frames per second (FPS) on the Raspberry Pi. This speed is sufficient for a real-time spraying system and confirms that the model is the most practical option for autonomous systems.

5. CONCLUSION AND FUTURE WORK

This study has successfully presented a specialized deep learning model that, optimized for working on edge computing devices with augmented images, has achieved high success in field tests for the detection and classification of pests, a critical component of autonomous farming systems. The application-specific model design, GAN-based data augmentation, and the importance of practical field tests have been demonstrated.

Although this proposed specialized CNN does not include current modules such as self-attention or depthwise separable convolutions, it has proven that performance stability in the field can minimize noise sensitivity with an advanced data augmentation technique without the need for architectural complexity. Experimental results show that deeper models like ResNet152 V2, despite advanced residual blocks, experience a decline in classification performance under adverse field conditions such as motion blur and partial occlusion. In contrast, with GAN-enhanced synthetic images, our simple 5-layer architecture has shown to produce more consistent results. This situation has proven that in autonomous systems with variable field conditions, a 'robust by design' simple architecture can perform better than 'accurate by design' complex models.

The table comparing his work with the literature is given in Table 3. When examining this table, models such as Tiny-LiteNet (98.6%) or DGS-YOLOv7-Tiny (mAP@0.5: 96.4%, 168 FPS) show very high accuracy values in a laboratory environment. However, the majority of these studies have been conducted on controlled datasets or in laboratory environments. In our study, the Raspberry Pi-based prototype was tested directly under field conditions. While the 82.5% accuracy achieved here may seem lower compared to large models like ResNet152V2, which achieves 92.4% accuracy in a laboratory environment, it demonstrates practical and reliable performance in a real agricultural application environment. In this regard, your work is one of the rare studies that combines academic accuracy rates with sustainability and practicality in the field. In addition, although some models (e.g., DGS-YOLOv7-Tiny, 4.43 M parameters and 10.2 GFLOPs) provide quite high accuracy, their hardware requirements are higher. However, the CNN we recommend is only 31 MB in size and can run at 7.5 FPS on a Raspberry Pi. While this value is lower compared to academic models that offer ultra-high FPS, it presents a solution that farmers working with limited hardware resources can directly access. This makes our work applicable on common and low-cost hardware instead of expensive GPUs or specialized accelerators.

A significant portion of the existing literature overlooks direct comparisons of inference latency (ms), focusing primarily on raw accuracy metrics. A stock Raspberry Pi 4B has a stable processing latency of 87 ms. Considering that small-scale autonomous agricultural sprayers typically operate at a ground speed of 0.5 to 1.5

m/s, a latency of less than 90 ms is quite reasonable.

Future work can be built upon the foundations of this project. Possible research directions include:

- **Real-Time Object Detection:** Integrating the current classification model with real-time object detection models like YOLO or SSD to determine not only the species of the insects but also their locations and numbers.
- **Life Cycle Analysis:** Expanding the dataset to include not only the adult stages of insects but also their different developmental stages such as egg, larva, and pupa.
- **Multi-Sensor Fusion:** Integrating different sensors such as thermal or hyperspectral cameras in addition to the RGB camera to enhance the system's detection capabilities.
- **Full Autonomous Platform Integration:** Developing an end-to-end system by mounting this "AI brain" onto an autonomous drone or a small ground vehicle for fully autonomous scanning and targeted spraying of large agricultural lands.

In conclusion, this study strongly demonstrates the potential of artificial intelligence to offer sustainable and technology-driven solutions to one of humanity's most fundamental problems, food security, and represents a significant step towards the future of smart agriculture technologies.

ACKNOWLEDGEMENT

One of the authors, Mahammade Karimzade, is grateful for the financial support provided by TÜBİTAK 2209-A - University Students Research Projects Support Program [Project No. 1919B012318079] to carry out this research study.

DECLARATIONS

Conflict of interest: No potential conflict of interest to declare by the authors.

REFERENCES

- [1] FAO, IFAD, UNICEF, WFP and WHO. "The State of Food Security and Nutrition in the World 2020. Transforming food systems for affordable healthy diets." Rome, FAO, 2020.
- [2] Nicolopoulou-Stamati P, Maipas S, Kotampasi C, Stamatis P, Hens L. Chemical Pesticides and Human Health: The Urgent Need for a New Concept in Agriculture. *Front Public Health*. 2016 Jul 18;4:148. doi: 10.3389/fpubh.2016.00148. PMID: 27486573; PMCID: PMC4947579.
- [3] R. Akhter and S. A. Sofi, 'Precision agriculture using IoT data analytics and machine learning', *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, Part B, pp. 5602–5618, 2022.
- [4] Chithambarathanu M, Jeyakumar MK. Survey on crop pest detection using deep learning and machine learning approaches. *Multimed Tools Appl*. 2023 Apr 11;1-34. doi: 10.1007/s11042-023-15221-3. Epub ahead of print. PMID: 37362671; PMCID: PMC10088765.
- [5] S. Wang et al., 'Advances in Deep Learning Applications for Plant Disease and Pest Detection: A Review', *Remote Sensing*, vol. 17, no. 4, 2025.
- [6] A. Junaidi et al., "Deep Learning and Edge Computing in Agriculture: A Comprehensive Review of Recent Trends and Innovations," in *IEEE Access*, vol. 13, pp. 137464-137490, 2025, doi: 10.1109/ACCESS.2025.3595621
- [7] Redmon, J., & Farhadi, A. "YOLOv3: An Incremental Improvement." *arXiv preprint arXiv:1804.02767*, 2018.
- [8] Nyakuri, J.P., Nkundineza, C., Gatera, O. et al. AI and IoT-powered edge device optimized for crop pest and disease detection. *Sci Rep* 15, 22905 (2025). <https://doi.org/10.1038/s41598-025-06452-5>
- [9] Yu, P., Zong, B., Geng, X. et al. DGS-Yolov7-Tiny: a lightweight pest and disease target detection model suitable for edge computing environments. *Sci Rep* 15, 29818 (2025). <https://doi.org/10.1038/s41598-025-13410-8>
- [10] Huang, Y., Liu, Z., Zhao, H., Tang, C., Liu, B., Li, Z., Wan, F., Qian, W., & Qiao, X. (2025). YOLO-YSTs: An Improved YOLOv10n-Based Method for Real-Time Field Pest Detection. *Agronomy*, 15(3), 575. <https://doi.org/10.3390/agronomy15030575>
- [11] Karam C, Awad M, Abou Jawdah Y, Ezzeddine N and Fardoun A (2022) GAN-based semi-automated augmentation online tool for agricultural pest detection: A case study on whiteflies. *Front. Plant Sci*. 13:813050. doi: 10.3389/fpls.2022.813050
- [12] Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., & Aila, T. "Training generative adversarial networks with limited data." **Advances in Neural Information Processing Systems**, 33, 12104-12114, 2020.
- [13] Simonyan, K., & Zisserman, A. "Very Deep Convolutional Networks for Large-Scale Image Recognition." *ICLR 2015*, 2015.
- [14] He, K., Zhang, X., Ren, S., & Sun, J. "Deep Residual Learning for Image Recognition." *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770-778, 2016.