digital

MDPI

Article

TinyML Classification for Agriculture Objects with ESP32

Danila Donskoy *, Valeria Gvindjiliya and Evgeniy Ivliev

Laboratory "Modeling and Development of Intelligent Agricultural Engineering Systems", Don State Technical University, Rostov-on-Don 344000, Russia; vvgvindjiliya@donstu.ru (V.G.); 123ivliev123@mail.ru (E.I.)
* Correspondence: dand22@bk.ru

Abstract

Using systems with machine learning technologies for process automation is a global trend in agriculture. However, implementing this technology comes with challenges, such as the need for a large amount of computing resources under conditions of limited energy consumption and the high cost of hardware for intelligent systems. This article presents the possibility of applying a modern ESP32 microcontroller platform in the agro-industrial sector to create intelligent devices based on the Internet of Things. CNN models are implemented based on the TensorFlow architecture in hardware and software solutions based on the ESP32 microcontroller from Espressif company to classify objects in crop fields. The purpose of this work is to create a hardware–software complex for local energy-efficient classification of images with support for IoT protocols. The results of this research allow for the automatic classification of field surfaces with the presence of "high attention" and optimal growth zones. This article shows that classification accuracy exceeding 87% can be achieved in small, energy-efficient systems, even for low-resolution images, depending on the CNN architecture and its quantization algorithm. The application of such technologies and methods of their optimization for energy-efficient devices, such as ESP32, will allow us to create an Intelligent Internet of Things network.

Keywords: microcontroller; machine learning; agroengineering; weed monitoring

check for updates

Academic Editor: Jorge Bernardino

Received: 17 July 2025
Revised: 13 September 2025
Accepted: 19 September 2025
Published: 2 October 2025

Citation: Donskoy, D.; Gvindjiliya, V.; Ivliev, E. TinyML Classification for Agriculture Objects with ESP32. *Digital* **2025**, *5*, 48. https://doi.org/10.3390/digital5040048

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Over the past decade, methods based on machine learning have intensively been under development in agriculture. This is due to the possibility of processing and analyzing big data on the state of crop areas, allowing a more complete look at the process of exploiting land resources and optimizing the efficiency of agricultural methods [1,2]. In response to a growing need for food security in countries, the use of hardware–software systems with integrated machine learning algorithms is especially relevant for effective monitoring of crop yields and crop area conditions [3–6].

Although modern methods of field monitoring are useful for crop forecasting and optimization of agricultural processes [7], there are limitations to the practical large-scale implementation of these systems. These include

- Insufficient spatial and temporal resolution of data [8];
- Incorrect use of neural network models [9];
- Poor reliability of systems in various operating conditions [10];
- Low energy efficiency [11].

The high cost of hardware, which is based on the use of NVIDIA Jetson platforms in many cases [12–16], affects the agricultural sector; although many promising AI tech-

nologies exist locally, they have different levels of implementation and efficiency. These reasons are slowing the growth of digital vision-based solutions for crop monitoring and implementing strategies to enhance food security and avert food crisis.

A review of existing systems for detecting and classifying objects of the agro-industrial complex in field conditions shows that most agrotechnical systems are redundant in terms of hardware and computing power. Currently, there is a large number of neural network models for solving problems in the agro-industrial complex. Most of these software solutions are built on resource-intensive architectures, such as YOLO [17,18], SSD [19], and Fast R-CNN [20], which can at least be launched on single-board computers such as NVIDIA Jetson and Raspberry Pi [21,22]. However, the cost of such solutions is quite high compared to microcontroller-based systems. Therefore, the development of new CNN architectures and their optimization for low-power computing devices have become an emerging trend in machine learning, known as Tiny Machine Learning (TinyML).

TinyML was pioneered by Google during the development of optimized speech recognition algorithms for deployment on mobile devices. Numerous research groups have recently begun adapting neural network models to the TinyML approach for various tasks in industrial sectors [23–25]. Research shows a detailed application of the TinyML approach with CNN optimization on an energy-efficient platform for the task of recognizing crops and weeds on agricultural land. Applied research in this specific area is scarce. The study most relevant to our work is that of Dennis Agyemanh Nana Gookyi et al. [26], who investigated the application of TinyML for corn leaf disease detection. The results suggest that this novel approach to plant classification could also enable the development of cost-effective, durable IoT devices for field monitoring, reducing both energy consumption and maintenance costs.

In addition, technologies based on the Internet of Things have great potential in agriculture [2,27]. It has been shown that Internet of Things technologies, in combination with the modern computing capabilities of microcontrollers, form a stable trend towards increasing the efficiency of data processing in environmental monitoring tasks based on devices with limited resources [28]. Therefore, this study presents an approach to creating a hardware and software complex for local energy-efficient classification of images in the agricultural sector with support for Internet of Things protocols. The results of classification of objects in agricultural fields are shown when working in scenarios with poor energy constraints and a low-quality dataset. The goal of this work is to analyze the benefits of energy-efficient deployment of machine learning in agricultural object classification tasks for a resource-constrained scenario.

This research uses the ESP32-CAM microcontroller for the hardware, with the resolution of images used to train the neural network ranging from 32 × 32 to 800 × 600 pixels. The work was carried out in several main stages. The first stage included preparing images of crop areas and images of plant crops and weeds. The second stage involved training the model to classify objects in the image. The third stage involves the transfer of model parameters for work in the microcontroller programming environment with subsequent quantization. Finally, the fourth stage involved testing and validating the resulting model on the microcontroller.

The main contributions of this work include the following:

1. A lightweight hardware and software complex for local energy-efficient image classification with support for Internet of Things protocols is presented.
2. An evaluation of the effectiveness of the TinyML model for the task of classifying agricultural objects based on low-resolution data ranging from 32 × 32 to 800 × 600 pixels is conducted.

3. The dependences of the dynamics of neural network training for data of different quality in a system with limited resources are shown.

The remainder of this paper is organized as follows: Section 2 presents the design stages of the neural network architecture and algorithms of the software package for classifying agricultural crop objects. The structure of the hardware part of the package is described. Section 3 presents the results of training two neural network architectures for a dataset with different resolutions. In Section 4, a conclusion is given on the possibility of using TinyML technologies in the agro-industrial sector for classifying objects in crop fields.

2. Materials and Methods

2.1. Consideration the Terms of Reference

When working with microcontroller systems, programmers have to deal with continuous optimization, especially if the problem to be solved is complex and requires a wide range of user functions at a minimum unit cost. This should be understood when creating realistic technical requirements for the system being developed.

Thus, the developed system should provide

1. The means to locally capture images of sufficient quality to be analyzed by a neural network.
2. Non-blocking code to asynchronously acquire data from output neurons and broadcast the data to the user.
3. A user interface displaying the captured image and its classification (using Wi-Fi or Bluetooth to communicate with the user).
4. The ability to detect three classes: plant-free zone, weed, and wheat. In classification, the classes "plant-free zone" and "weed" will belong to an enlarged classification for the user-"attention zone." Thus, support for neural computing with CNN architecture is required. The classification accuracy should not be less than 90%, considering the computational power of the microcontroller.
5. A subroutine for local accumulation of the dataset, which will allow us to generate unique data for additional training of the applied neural model.

2.2. Definition of Hardware and Software Complex

The ESP32-CAM microcontroller platform was chosen for debugging of the software system. It consists of the ESP32 microcontroller itself, which has two physical 32-bit cores on Xtensa LX6 architecture. The board also includes a 4 MB PSRAM chip to relieve the controller's memory when capturing high-quality images with resolutions higher than 800 × 600 pixels, which greatly improves the system's performance and response. Support for an external MicroSD card is provided for data storage. The computational module was chosen after evaluating hardware platforms with superior performance to the ESP32 microcontroller. A review of the research literature confirmed the effectiveness of the ESP32-CAM for object recognition and classification tasks [29]. The low weight (less than 10 g) and low power consumption (which in peaks does not exceed 500 mA) of this hardware platform allows for integration of the system both on wheeled and tracked vehicles, as well as on aircraft (e.g., quadrocopters) for analyzing images from the air. Furthermore, market analysis also demonstrated its economic viability, as no alternative board provides higher performance for under USD 10. Thus, the ESP32-CAM platform was adopted for this applied study.

The basic camera for this platform is the OV2640, which enables JPEG images with a maximum resolution of 1600 × 1200 pixels to be captured with a capture angle of 66°. The camera provides the following output formats: YUV(422/420)/YCbCr422, 8-bit compressed data, RGB565/555, and 8-/10-bit Raw RGB data. This project uses the 8-bit RGB format. The image resolution and color reproduction quality are sufficient for image analysis. These

modules are often used in conjunction with more powerful single-board computers, such as the Raspberry Pi [30], which enable future computational power scaling without any modifications to the optical hardware's specifications or properties.

When integrating the system as an attachment for agricultural machinery, synchronization of multiple devices during system operation in the field can be implemented based on the ESP-MESH protocol over Wi-Fi. This ensures accurate periods of photo fixation and collection of information from multiple nodes.

This layout is used in tandem with the popular GPS/GLONASS module Beitian BN-220. This is necessary to add information on photo-fixation coordinates to the collected data arrays. Then, the information is entered into the mapping system, which allows for a basic analysis to be performed after data collection and for a heat map of the attention zones of the cultivated areas to be created. This system is not this paper's object of research, but a continuation of the formed hardware–software complex of weediness analysis of sown areas with the use of artificial intelligence algorithms. This study is expected to allow users to conduct this work more efficiently and purposefully in future visits to the fields.

For the transition to the program complex, it is necessary to collect data arrays for each analyzed class of images. Data was collected using various types of field equipment ranging from a height of 0.8 to 2 m above the ground using an ESP32-CAM-based device, the schematic diagram of which is shown in Figure 1.

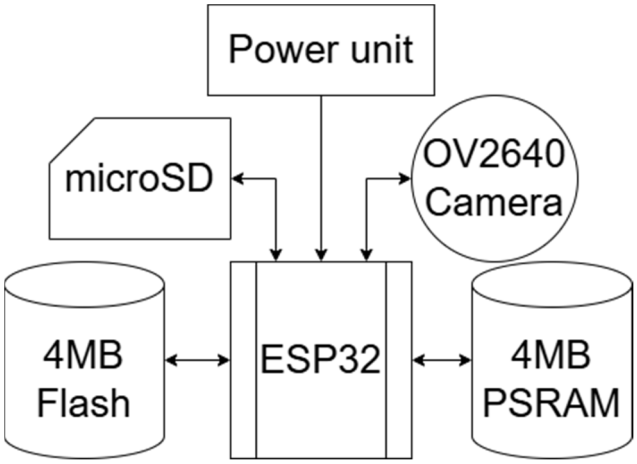


Figure 1. Schematic diagram of the data acquisition device.

This layout was also used to debug image classification algorithms. This layout was powered either from a step-down converter with a power ranging from 24–12 V to 5 V in a moving vehicle or from a USB-TTL converter based on the CH340 chip.

As a result, several hundred unique photos were collected, divided by the classes described above. However, a preliminary review of the obtained data showed their low diversity and the need for refinement by increasing their base to find characteristic features. Therefore, we decided to take advantage of existing datasets used for deep learning in larger projects. As a result, the following datasets with information useful for solving our problem were partially considered and applied [31–36], and the total number of photos before augmentation was 12,272.

As TinyML algorithms improve, we plan to release competing microcontroller software to perform real-time (or close to it, so that the processing time is less than 2 s) weed classification for point field processing. This will change approaches to the engineering of equipment like WeedSeeker 2 [37] and, most importantly, dramatically reduce their power consumption and cost.

Engineering of the neural network architecture and algorithms of the program complex was carried out in several stages:

1. Extraction and classification of data samples from external datasets for subsequent use in neural network training;
2. Preprocessing and augmentation of data, including resizing to multiple resolutions ranging from 512 × 512 to 80 × 60 pixels;
3. Construction of training, testing, and validation datasets;
4. Visual inspection of compiled datasets to ensure correctness and consistency;
5. Definition and implementation of the neural network architecture using TensorFlow, as well as model compilation and training;
6. Analysis of training dynamics and evaluation of prediction accuracy;
7. Selection of representative datasets, followed by model optimization and quantization for deployment efficiency;
8. Assessment of the accuracy of the optimized model;
9. Conversion of trained model parameters into a binary format suitable for use in embedded C++ environments;
10. Development of microcontroller firmware supporting TensorFlow Lite [38] and implementation of a basic web interface for interaction;
11. Integration of the trained neural network into the firmware codebase;
12. Testing of image classification performance on a physical hardware prototype.

An example of a random dataset after generating class datasets with a resolution of 160 × 120 pixels is shown in Figure 2.

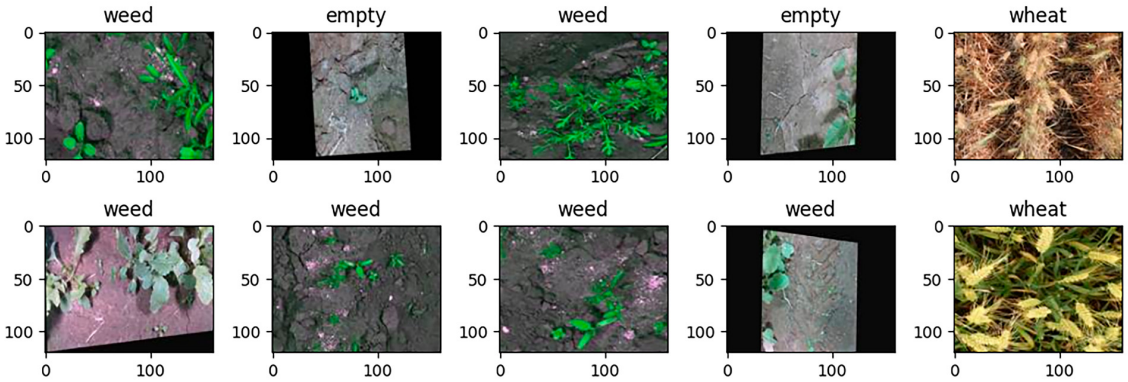


Figure 2. An example of the different classes in the dataset.

The resolution of the image fed to the neural network affects the number of features that will be found.

A representative sample was formed and then the model was quantized, and the transition to an eight-bit integer data type was carried out. This means that the input and output values are now scaled to values ranging from −128 to +127. After that, the already quantized model of the developed program was ported to the ESP32 microcontroller.

We also noticed that, with input data of low resolution (below 100 × 100 pixels), quantization of trained models leads to critical loss of classification accuracy. Thus, if a normal model gave 69–71% classification accuracy, the accuracy decreased to 50–54% after quantization.

When trained on datasets of a higher resolution, such as those ranging from 256 × 256 pixels up to 800 × 600 pixels, the classification accuracy improved but, when trying to run these models on a microcontroller, the required memory stack exceeded the existing free space. An example of memory allocation requests at different image resolutions is presented below (Figure 3). It should be noted that the microcontroller platform used has only 4 MB of memory, both for the main program and for the neural network computation stack.

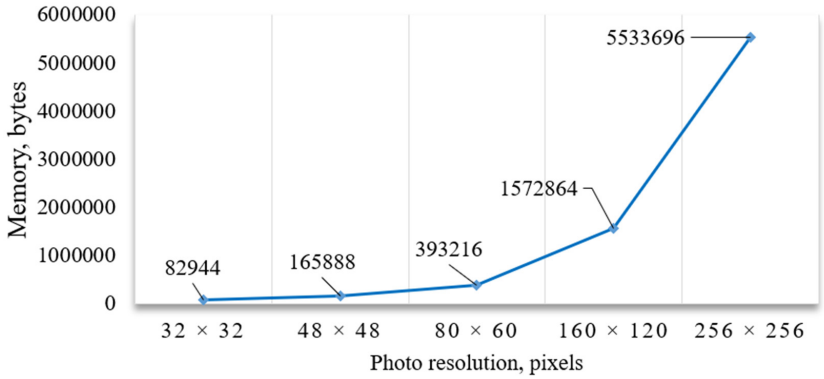


Figure 3. An example of the dependence of the average required amount of memory on the analyzed photos' resolution for one of the CNN architectures used.

It should be noted that the amount of memory needed for microcontroller systems is an important parameter to consider when selecting the architecture of a neural network, since any changes in the architecture will affect the required memory stack and the classification time (Figure 4). The following two convolutional neural network architectures (Figures 5 and 6) were chosen for this research. Figure 3 shows the characteristics of the first architecture (Figure 5).

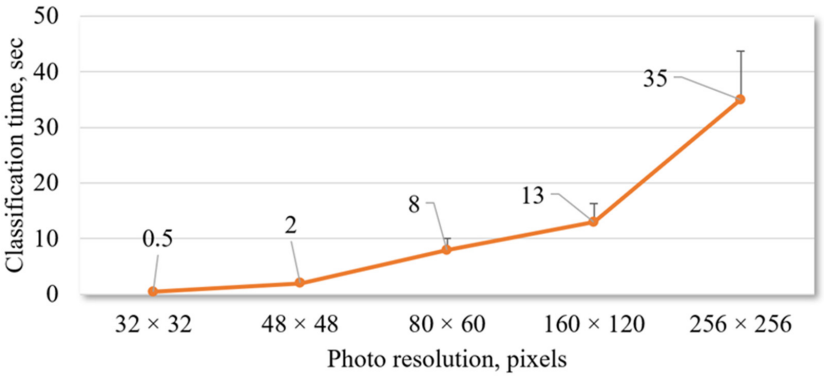


Figure 4. The dependence of the classification time on the resolution of the photo.

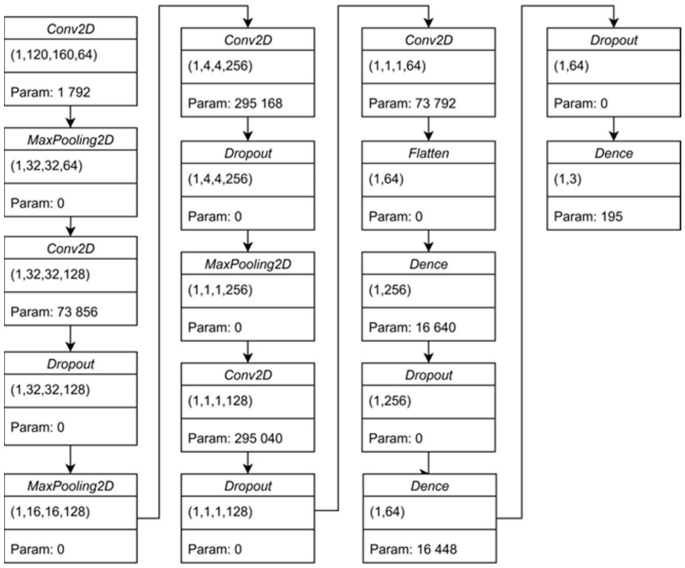


Figure 5. The first CNN architecture. Total params: 838,723 (3.20 MB).

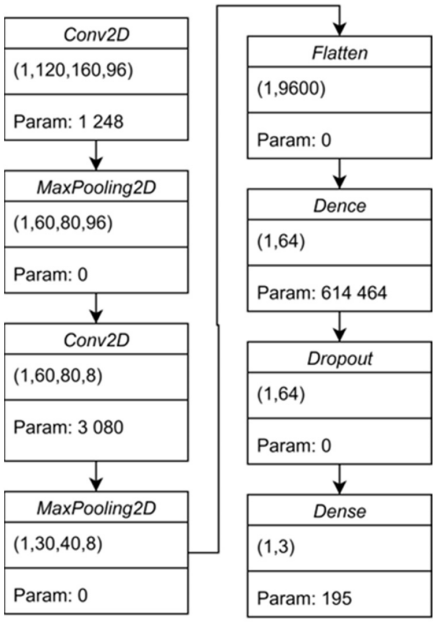


Figure 6. The second CNN architecture. Total params: 618,987 (2.36 MB).

The first architecture was applied in one of the existing projects aiming to classify visually described spectrograms after windowed Fourier transform for sound classification based on an ESP32 microcontroller [39].

The second architecture was obtained by investigating the learning dynamics of different neural network architectures and their resulting accuracy.

Several versions of the microcontroller program were tested. The first version was based on the classical library for microcontrollers, "tflite-micro" [38], and the second version was based on "ArduTFLite" [40]. Since no differences were observed in the performance of tasks, the program based on "ArduTFLite" was determined to be the main one to increase the readability of the code. The structure of the program is based on previously conducted research on the classification of audio recording spectrograms [39,41].

3. Results and Discussion

Datasets with different training resolutions were investigated. At resolutions lower than 70 × 70 pixels, the features that can be used to separate photos visually disappear or become implicit, so we decided to train the models at higher resolutions (Figure 7).

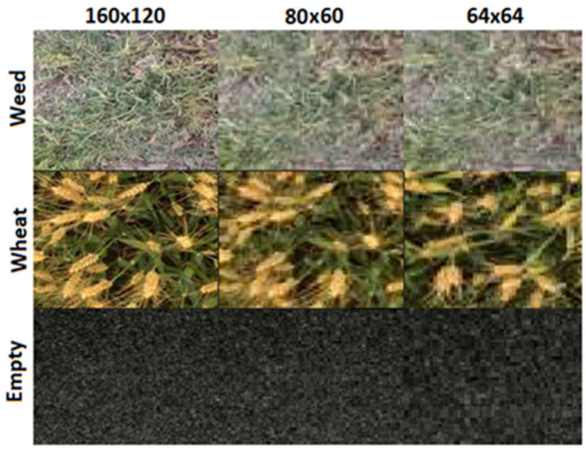


Figure 7. An example of data with different resolutions.

The standard metrics of the TensorFlow package were used to describe the learning processes of the models. More precisely, the "categorical_accuracy" method was applied to evaluate accuracy:

$$Accuracy = \frac{1}{n}\sum_{i-1}^{n}(y_i == \hat{y}_i) \quad (1)$$

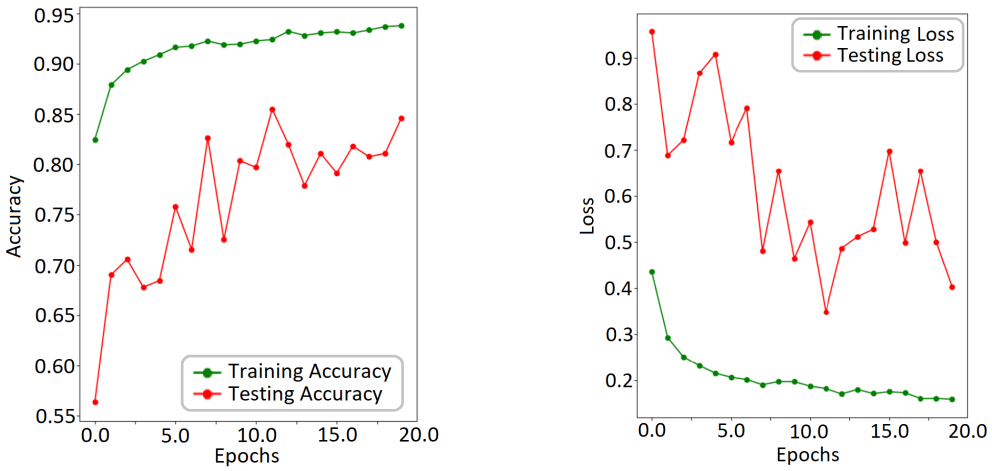
where y_i is the true class label, $\hat{y}_i$ is the predicted class label, and n is the total number of samples.

To calculate losses, the "categorical_crossentropy" method was used:

$$Loss = -\sum_{i-1}^{n}(y_i \cdot log(\hat{y}_i)) \quad (2)$$

The resulting metrics are shown in the figures below. "Training Accuracy" is a metric that determines how well a model predicts the correct class in a photo from a training dataset. For example, a score of 90% or higher means that the model has memorized the images from the training sample well. "Testing Accuracy" determines how accurately the model predicts the correct class based on a validation dataset that it has never considered in training. For example, a score of 10% means that when submitting a new image, the model does not recognize what it is. "Training Loss" determines the number of model errors on the training data, in numerical terms, through the loss function. "Testing Loss" reveals how many mistakes the model makes on the test data; a drop in this value means that the model is learning to understand the essence (key features of classes) and is not just memorizing images.

Figure 8 illustrates the training dynamics of the convolutional neural network (CNN) based on the architecture presented in Figure 5. This architecture demonstrated suboptimal performance, particularly at higher image resolutions, which led to the decision to adopt an alternative model, as shown in Figure 6. Training with the second architecture proved to be more efficient and yielded significantly better results, as reflected in the training graphs presented in Figures 9 and 10.



(**a**) (**b**)

Figure 8. Neural network training at a resolution of 64 × 64 pixels in the input image: (**a**) accuracy; (**b**) loss.

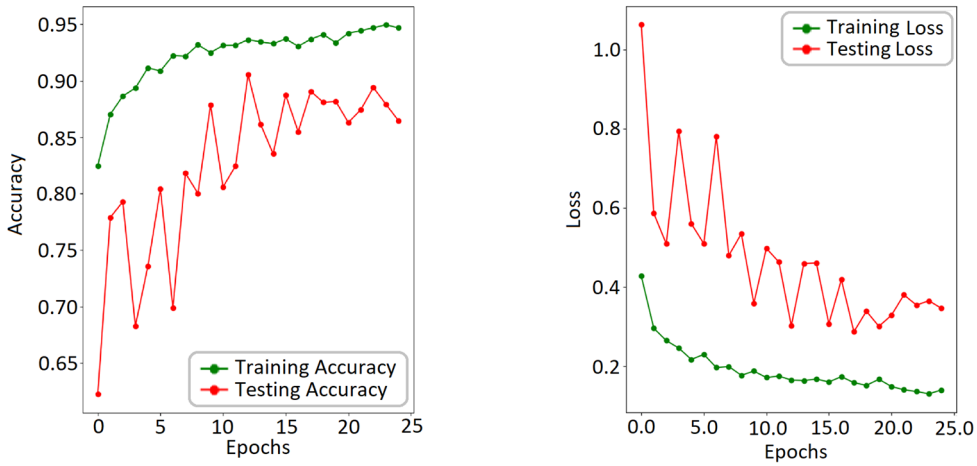


Figure 9. Neural network training at a resolution of 80 × 60 pixels in the input image: (**a**) accuracy; (**b**) loss.

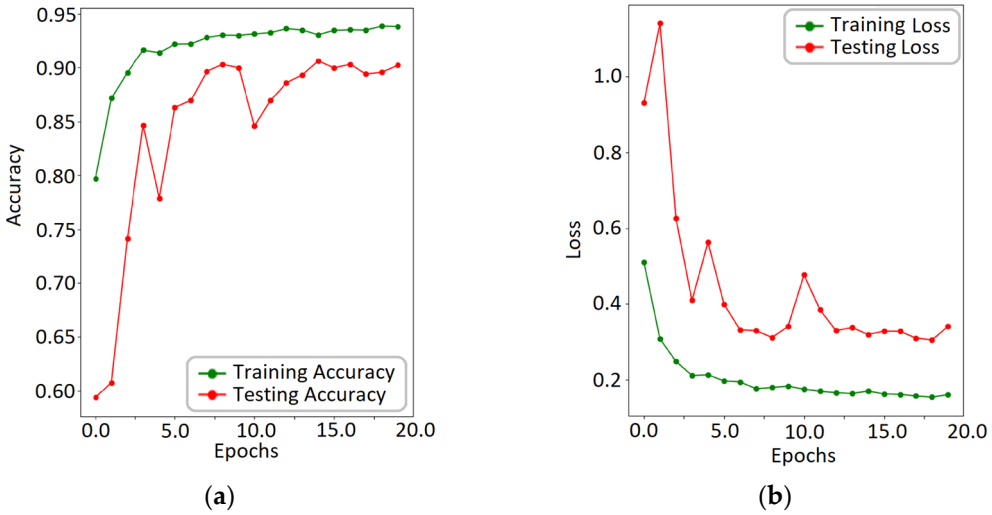


Figure 10. Neural network training at a resolution of 160 × 120 pixels in the input image: (**a**) accuracy; (**b**) loss.

The resulting classification accuracies for both CNN architectures—standard and quantized—at various input resolutions are summarized in Table 1. The quantized version of the second architecture (Figure 6) achieved the highest accuracy across all resolutions, including the lowest resolution of 80 × 60 pixels.

Table 1. Classification accuracy at different resolutions of input data.

Model	Quantized	Accuracy (320 × 240), %	Accuracy (160 × 120), %	Accuracy (80 × 60), %
The first CNN architecture	No	82.5	79.32	74.25
The first CNN architecture	Yes	76.6	71.43	60.98
The second CNN architecture	No	94.83	90.21	84.58
The second CNN architecture	Yes	96.15	92.86	87.50

An estimate of the memory consumption for each CNN configuration when deployed on the ESP32 microcontroller is provided in Table 2.

Table 2. Memory usage of the CNN models on ESP32.

Model	Memory (320 × 240), KB	Memory (160 × 120), KB	Memory (80 × 60), KB
The first CNN architecture	3584	1536	384
The second CNN architecture	5120	2304	562

Figure 11 shows an example of image classification performed using the second CNN architecture. The results are presented as pairs of ground truth and predicted classes, allowing for a direct comparison between the expected and actual outputs of the trained model.

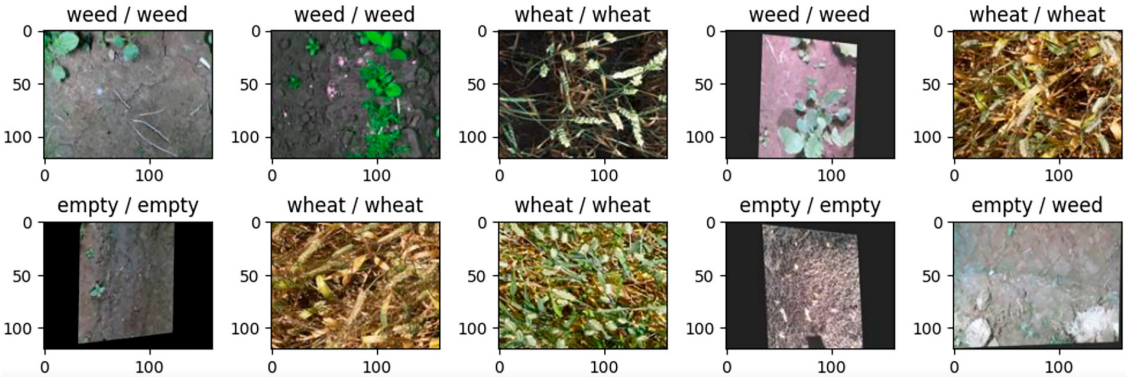


Figure 11. Classification results obtained using the second CNN architecture (ground truth/predicted class).

The classification results of the finished system with an integrated neural network are shown in the figure below. The microcontroller initializes a Wi-Fi access point with a static IP address for quick connection to a local HTTP server with WebSocket support. When a user clicks on the "classification" button, the controller captures the image, and from 4 to 15 s, it generates a result with a distribution of classes from −128 to 127. This result is then stored in an external MicroSD card. The response is sent to the user via WebSocket in the form of an array with integer values of the "byte" type (Figures 12 and 13).

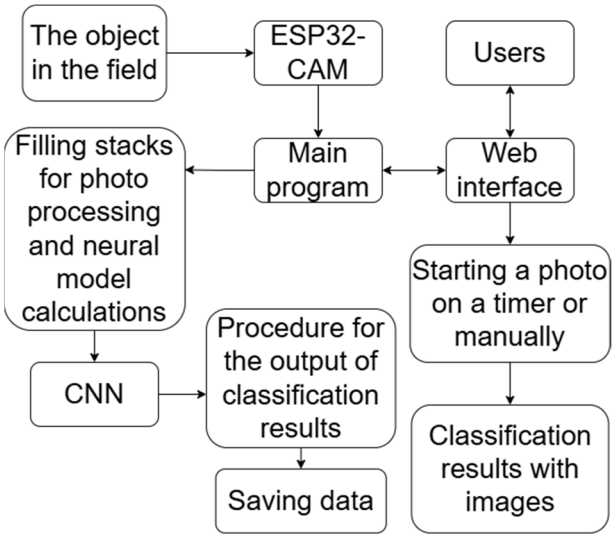


Figure 12. Functional block diagram of the system.

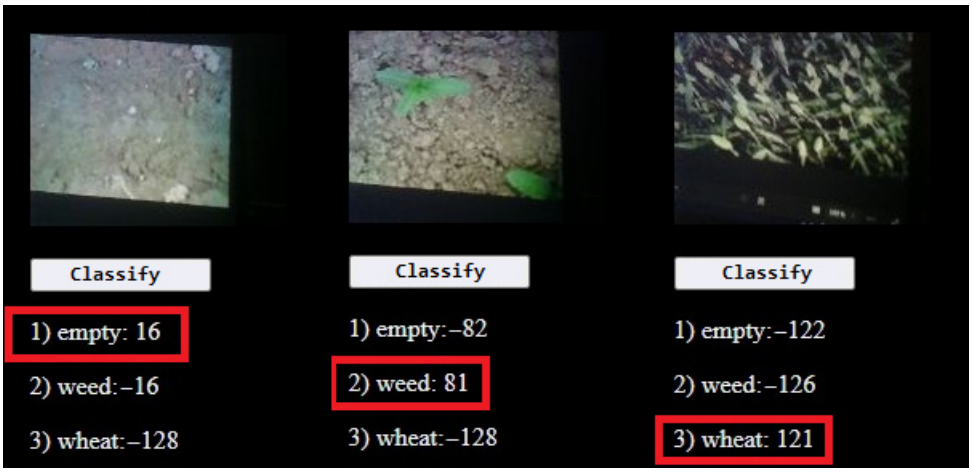


Figure 13. An example of classification on microcontroller via web interface.

The obtained results demonstrate the feasibility of classifying plant objects in the field without relying on high-performance computing equipment or prolonged data processing time, particularly during CNN training, which currently remains a limitation for real-time monitoring systems [42]. The accuracy of the results depends on the quality of training images for each data class; however, classification accuracy exceeding 87% can be achieved even for low-resolution images depending on the CNN architecture and its quantization algorithm (Table 1). These findings are inferior to solutions based on the NVIDIA Jetson Nano platform. In the work by Kaldarova M. et al. [15], the training accuracy of the weed recognition model reaches 0.98 with an inference time of 0.2 s. The model was trained using images with a resolution of 500 × 350 pixels, which is six times higher than the image resolution used in the present study. This factor also impacts recognition quality and performance hardware. Finally, the superior results reported in [15] were achieved with significantly higher power consumption (typically 5–15 W compared to 1 W for the ESP32) and substantially greater hardware cost (approximately USD 200 compared to less than USD 10 for the ESP32). The optimal solution depends on the specific requirements for power availability, scalability, and computational demand. Our work demonstrates that for plant object classification in agricultural fields, microcontroller-tier hardware is not only viable but optimal when energy efficiency and cost are the primary constraints.

Thus, even when implemented on low-power microcontrollers, the CNN enables plant object classification across crop fields while reducing the energy consumption of field monitoring hardware–software systems without compromising accuracy.

4. Conclusions

This article investigated the possibility of using TinyML technologies in the agro-industrial sector to classify objects in crop fields. We used a CNN in a low-power microcontroller, which showed good accuracy in determining classes of data with different quality.

As a result, the architecture used in practice was chosen (Figure 6) and the algorithm operated at a resolution of 80 × 60, which ensured an accuracy of 87% with a maximum classification time of up to 8 s. Data collection and classification are implemented locally on the micro-controller, which allows for mobile analysis of the surfaces of crop fields and the creation of data arrays on their condition. We can also plot a map showing the weediness and crop yields of fields. Implementing such a system can help to optimize the use of resources; for example, when treating fields with fertilizers and herbicides.

The results of this study can be used to create databases with a pre-classified set of photos of a crop area with a set of geographic coordinates on the field for the cultivation of

grain crops. The system developed for classifying plant objects based on a microcontroller and the TinyML approach will be used to create maps of field weediness to determine areas resistant to herbicides.

Further research will focus on conducting a deeper analysis of the TinyML approach in classifying plant objects on fields. We will consider applying hybrid KAN-MIXER and KAN-BiLSTM architectures that surpass traditional CNNs in both data interpretability and feature extraction efficiency [43,44]. The adoption of these models will be driven by the next phase of our research, which focuses on classifying more complex datasets to develop integrated systems for agricultural field mapping. The result of the considered neural network deployed on a microcontroller will be compared with the most prominent network architectures deployed on the NVIDIA Jetson platform. We will also prioritize integrating the TinyML approach with Internet of Things systems to ensure the best solution to the problem of classifying objects in real time and to minimize computing resources.

Author Contributions: Conceptualization, D.D. and V.G.; methodology, D.D. and V.G.; software, D.D.; validation, E.I.; data curation D.D.; writing—original draft preparation, D.D.; writing—review and editing, V.G.; visualization, E.I. All authors have read and agreed to the published version of the manuscript.

Funding: This work was carried out within the framework of the project "Mathematical modeling and algorithms for modeling plant growth based on an automated cartographic system" (FZNE2024-0006).

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

TinyML	Tiny Machine Learning
CNN	Convolutional Neural Network
IoT	Internet of Things
PSRAM	Pseudo Static Random Access Memory
GPS	Global Positioning System
GLONASS	Global Navigation Satellite System

References

1. Adisa, O.; Ilugbusi, B.S.; Adewunmi, O.; Franca, O.; Ndubuisi, L. A comprehensive review of redefining agricultural economics for sustainable development: Overcoming challenges and seizing opportunities in a changing world. *World J. Adv. Res. Rev.* **2024**, *21*, 2329–2341. [CrossRef]
2. Kumar, V.; Sharma, K.V.; Kedam, N.; Patel, A.; Kate, T.R.; Rathnayake, U. A comprehensive review on smart and sustainable agriculture using IoT technologies. *Smart Agric. Technol.* **2024**, *8*, 100487. [CrossRef]
3. Zhang, C.; Di, L.; Lin, L.; Zhao, H.; Li, H.; Yang, A.; Yang, Z. Cyberinformatics tool for in-season crop-specific land cover monitoring: Design, implementation, and applications of iCrop. *Comput. Electron. Agric.* **2023**, *213*, 108199. [CrossRef]
4. Anam, I.; Arafat, N.; Hafiz, M.S.; Jim, J.R.; Kabir, M.M.; Mridha, M.F. A systematic review of UAV and AI integration for targeted disease detection, weed management, and pest control in precision agriculture. *Smart Agric. Technol.* **2024**, *9*, 100647. [CrossRef]
5. Wang, A.; Zhang, W.; Wei, X. A review on weed detection using ground-based machine vision and image processing techniques. *Comput. Electron. Agric.* **2019**, *158*, 226–240. [CrossRef]
6. Kanning, M.; Kühling, I.; Trautz, D.; Jarmer, T. High-resolution UAV-based hyperspectral imagery for LAI and chlorophyll estimations from wheat for yield prediction. *Remote Sens.* **2018**, *10*, 2000. [CrossRef]
7. Waqas, M.; Naseem, A.; Humphries, U.W.; Hlaing, P.T.; Dechpichai, P.; Wangwongchai, A. Applications of machine learning and deep learning in agriculture: A comprehensive review. *Green Technol. Sustain.* **2025**, *3*, 100199. [CrossRef]
8. Lacerda, C.F.; Ampatzidis, Y.; Neto, A.D.O.C.; Partel, V. Cost-efficient high-resolution monitoring for specialty crops using AgI-GAN and AI-driven analytics. *Comput. Electron. Agric.* **2025**, *237*, 110678. [CrossRef]

9. Castillo-Girones, S.; Munera, S.; Martínez-Sober, M.; Blasco, J.; Cubero, S.; Gómez-Sanchis, J. Artificial Neural Networks in Agriculture, the core of artificial intelligence: What, When, and Why. *Comput. Electron. Agric.* **2025**, *230*, 109938. [CrossRef]
10. Dhanush, G.; Khatri, N.; Kumar, S.; Shukla, P.K. A comprehensive review of machine vision systems and artificial intelligence algorithms for the detection and harvesting of agricultural produce. *Sci. Afr.* **2023**, *21*, e01798. [CrossRef]
11. Paris, B.; Vandorou, F.; Balafoutis, A.T.; Vaiopoulos, K.; Kyriakarakos, G.; Manolakos, D.; Papadakis, G. Energy use in open-field agriculture in the EU: A critical review recommending energy efficiency measures and renewable energy sources adoption. *Renew. Sustain. Energy Rev.* **2022**, *158*, 112098. [CrossRef]
12. Sunil, G.C.; Upadhyay, A.; Sun, X. Development of software interface for AI-driven weed control in robotic vehicles, with time-based evaluation in indoor and field settings. *Smart Agric. Technol.* **2024**, *9*, 100678. [CrossRef]
13. Kariyanna, B.; Sowjanya, M. Unravelling the use of artificial intelligence in management of insect pests. *Smart Agric. Technol.* **2024**, *8*, 100517. [CrossRef]
14. Kuznetsov, P.; Kotelnikov, D.; Voronin, D.; Evstigneev, V.; Yakimovich, B.; Kelemen, M. Intelligent monitoring of the physio-logical state of agricultural products using UAV. *MM Sci. J.* **2024**, *2024*, 7772–7781. [CrossRef]
15. Kaldarova, M.; Akanova, A.; Nazyrova, A.; Mukanova, A.; Tynykulova, A. Identification of weeds in fields based on computer vision technology. *East.-Eur. J. Enterp. Technol.* **2023**, *4*, 44–52. [CrossRef]
16. Valladares, S.; Toscano, M.; Tufiño, R.; Morillo, P.; Vallejo-Huanga, D. Performance Evaluation of the Nvidia Jetson Nano Through a Real-Time Machine Learning Application. *Intell. Hum. Syst. Integr.* **2021**, *1322*, 343–349. [CrossRef]
17. Mukhamediev, R.I.; Smurygin, V.; Symagulov, A.; Kuchin, Y.; Popova, Y.; Abdoldina, F.; Tabynbayeva, L.; Gopejenko, V.; Oxenenko, A. Fast Detection of Plants in Soybean Fields Using UAVs, YOLOv8x Framework, and Image Segmentation. *Drones* **2025**, *9*, 547. [CrossRef]
18. Gao, X.; Wang, G.; Qi, J.; Wang, Q.; Xiang, M.; Song, K.; Zhou, Z. Improved YOLO v7 for Sustainable Agriculture Significantly Improves Precision Rate for Chinese Cabbage (*Brassica pekinensis* Rupr.) Seedling Belt (CCSB) Detection. *Sustainability* **2024**, *16*, 4759. [CrossRef]
19. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot Multibox Detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; Volume 1, pp. 21–37. [CrossRef]
20. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Neural Inf. Process. Syst.* **2015**, *28*, 1137–1149. [CrossRef]
21. Brunell, D.; Albanese, A.; d'Acunto, D.; Nardello, M. Energy neutral machine learning based IoT device for pest detection in precision agriculture. *IEEE Internet Things Mag.* **2019**, *2*, 10–13. [CrossRef]
22. Ivliev, E.; Demchenko, V.; Obukhov, P. Automatic Monitoring of Smart Greenhouse Parameters and Detection of Plant Diseases by Neural Networks. In *Robotics, Machinery and Engineering Technology for Precision Agriculture*; Shamtsyan, M., Pasetti, M., Beskopylny, A., Eds.; Smart Innovation, Systems and Technologies; Springer: Singapore, 2022; Volume 247, pp. 37–45. [CrossRef]
23. Langer, T.; Widra, M.; Beyer, V. TinyML Towards Industry 4.0: Resource-Efficient Process Monitoring of a Milling Machine. *arXiv* **2025**, arXiv:2508.16553.
24. Vu, T.H.; Tu, N.H.; Huynh-The, T.; Lee, K.; Kim, S.; Voznak, M.; Pham, Q.V. Integration of TinyML and LargeML: A Survey of 6G and Beyond. *arXiv* **2025**, arXiv:2505.15854.
25. Dockendorf, C.; Mitra, A.; Mohanty, S.P.; Kougianos, E. Lite-Agro: Exploring Light-Duty Computing Platforms for IoAT-Edge AI in Plant Disease Identification. In *Proceedings of the IFIP International Internet of Things Conference, Denton, TX, USA, 2–3 November 2023*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 371–380. [CrossRef]
26. Gookyi, D.; Wulnye, F.; Arthur, E.; Ahiadormey, R.; Agyemang, J.; Aguekum, K.; Gyaang, R. TinyML for Smart Agriculture: Comparative Analysis of TinyML Platforms and Practical Deployment for Maize Leaf Disease Identification. *Smart Agric. Technol.* **2024**, *8*, 100490. [CrossRef]
27. Choudhary, V.; Guha, P.; Pau, G.; Mishra, S. An overview of smart agriculture using internet of things (IoT) and web services. *Environ. Sustain. Indic.* **2025**, *26*, 100607. [CrossRef]
28. Sabovic, A.; Fontaine, J.; De Poorter, E.; Famaey, J. Energy-aware tinyML model selection on zero energy devices. *Internet Things* **2025**, *30*, 101488. [CrossRef]
29. Sumari, A.; Annurroni, I.; Ayuningtyas, A. The Internet-of-Things-based Fishpond Security System Using NodeMCU ESP32-CAM Microcontroller. *J. RESTI Rekayasa Sist. Dan Teknol. Inf.* **2025**, *9*, 51–61. [CrossRef]
30. Adi, P.D.P.; Wahyu, Y. Performance evaluation of ESP32 Camera Face Recognition for various projects. *Internet Things Artif. Intell. J.* **2022**, *2*, 10–21. [CrossRef]
31. Panara, U.; Pandya, R.; Rayja, M. Crop and Weed Detection Data with Bounding Boxes. 2020. Available online: https://www.kaggle.com/datasets/ravirajsinh45/crop-and-weed-detection-data-with-bounding-boxes (accessed on 6 June 2025).

32. Steininger, D.; Trondl, A.; Croonen, G.; Simon, J.; Widhalm, V. The CropAndWeed Dataset: A Multi-Modal Learning Approach for Efficient Crop and Weed Manipulation. In Proceedings of the 2023 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 3–7 January 2023.
33. Haug, S.; Ostermann, J. A Crop/Weed Field Image Dataset for the Evaluation of Computer Vision Based Precision Agriculture Tasks. In Proceedings of the Computer Vision—ECCV 2014 Workshops, Zurich, Switzerland, 6, 7 and 12 September 2014.
34. Lameski, P. Weed-Datasets. Available online: https://github.com/zhangchuanyin/weed-datasets?ysclid=m9b389hlew273226771 (accessed on 6 June 2025).
35. David, E.; Madec, S.; Sadeghi-Tehran, P.; Aasen, H.; Zheng, B.; Liu, S.; Kirchgessner, N.; Ishikawa, G.; Nagasawa, K.; Badhon, M.A. Global Wheat Head Detection (GWHD) dataset: A large and diverse dataset of high-resolution RGB-labelled images to develop and benchmark wheat head detection methods. *Sci. Partn. J.* **2020**, *2020*, 1–15. [CrossRef]
36. Olsen, A.; Konovalov, D.A.; Philippa, B. DeepWeeds: A Multiclass Weed Species Image Dataset for Deep Learning. *Sci. Rep.* **2019**, *9*, 2058. [CrossRef]
37. WeedSeeker 2 Spot Spraying System. Available online: https://ru.ptxtrimble.com/product/sistema-tochechnogo-opriskivania-weedseeker2/ (accessed on 6 June 2025).
38. TensorFlow Lite for Microcontrollers (Tflite-Micro). Available online: https://github.com/tensorflow/tflite-micro/tree/main (accessed on 6 June 2025).
39. Donskoy, D.Y.; Lukyanov, A.D. Implementation of neural networks in IoT based on ESP32 microcontrollers. In Proceedings of the XVII International Scientific and Technical Conference «Dynamics of Technical Systems» (DTS-2021), Rostov-on-Don, Russia, 9–11 September 2021.
40. Arduino-Style TensorFlow Lite Micro Library (ArduTFLite). Available online: https://github.com/spaziochirale/ArduTFLite (accessed on 6 June 2025).
41. Rudoy, D.V.; Chigvintsev, V.V.; Olshevskaya, A.V. Use of neural networks for agrochemical analysis of soil. In Proceedings of the IV International Forum «Youth in Agribusiness», Rostov-on-Don, Russia, 5–8 November 2024.
42. Rawat, W.; Wang, Z. Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. *Neural Comput.* **2017**, *29*, 2352–2449. [CrossRef] [PubMed]
43. Chechkin, A.; Pleshakova, E.; Gataullin, S. A Hybrid KAN-BiLSTM Transformer with Multi-Domain Dynamic Attention Model for Cybersecurity. *Technologies* **2025**, *13*, 223. [CrossRef]
44. Jamali, A.; Roy, S.K.; Hong, D.; Lu, B.; Ghamisi, P. How to Learn More? Exploring Kolmogorov–Arnold Networks for Hyperspectral Image Classification. *Remote Sens.* **2024**, *16*, 4015. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.