12-2023

AICropCAM: Deploying classification, segmentation, detection, and counting deep-learning models for crop monitoring on the edge

Nipuna Chamara
nabeysingheherathm2@huskers.unl.edu

Geng (Frank) Bai
University of Nebraska - Lincoln, gbai2@unl.edu

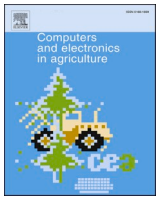
Yufeng Ge
University of Nebraska - Lincoln, yge2@unl.edu

Computers and Electronics in Agriculture 215 (2023) 108420

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag



AICropCAM: Deploying classification, segmentation, detection, and counting deep-learning models for crop monitoring on the edge

Nipuna Chamara [a], Geng Bai [a], Yufeng Ge [a,b,*]

[a] Department of Biological Systems Engineering, University of Nebraska-Lincoln, Lincoln, NE 68583, USA
[b] Center for Plant Science Innovation, University of Nebraska-Lincoln, Lincoln, NE 68588, USA

ARTICLE INFO

Keywords:
Artificial intelligence
Computer vision
Edge computing
Internet of things
LoRaWAN
Precision agriculture

ABSTRACT

Precision Agriculture (PA) promises to meet the future demands for food, feed, fiber, and fuel while keeping their production sustainable and environmentally friendly. PA relies heavily on sensing technologies to inform site-specific decision supports for planting, irrigation, fertilization, spraying, and harvesting. Traditional point-based sensors enjoy small data sizes but are limited in their capacity to measure plant and canopy parameters. On the other hand, imaging sensors can be powerful in measuring a wide range of these parameters, especially when coupled with Artificial Intelligence. The challenge, however, is the lack of computing, electric power, and connectivity infrastructure in agricultural fields, preventing the full utilization of imaging sensors. This paper reported AICropCAM, a field-deployable imaging framework that integrated edge image processing, Internet of Things (IoT), and LoRaWAN for low-power, long-range communication. The core component of AICropCAM is a stack of four Deep Convolutional Neural Networks (DCNN) models running sequentially: CropClassiNet for crop type classification, CanopySegNet for canopy cover quantification, PlantCountNet for plant and weed counting, and InsectNet for insect identification. These DCNN models were trained and tested with >43,000 field crop images collected offline. AICropCAM was embodied on a distributed wireless sensor network with its sensor node consisting of an RGB camera for image acquisition, a Raspberry Pi 4B single-board computer for edge image processing, and an Arduino MKR1310 for LoRa communication and power management. Our testing showed that the time to run the DCNN models ranged from 0.20 s for InsectNet to 20.20 s for CanopySegNet, and power consumption ranged from 3.68 W for InsectNet to 5.83 W for CanopySegNet. The classification model CropClassiNet reported 94.5 % accuracy, and the segmentation model CanopySegNet reported 92.83 % accuracy. The two object detection models PlantCountNet and InsectNet reported mean average precision of 0.69 and 0.02 for the test images. Predictions from the DCNN models were transmitted to the ThingSpeak IoT platform for visualization and analytics. We concluded that AICropCAM successfully implemented image processing on the edge, drastically reduced the amount of data being transmitted, and could satisfy the real-time need for decision-making in PA. AICropCAM can be deployed on moving platforms such as center pivots or drones to increase its spatial coverage and resolution to support crop monitoring and field operations.

1. Introduction

The demands for food, feed, fiber, and fuel increase rapidly due to the fast expansion of the global population, income growth, technological advancement, and transport and logistics improvements (van Dijk et al., 2021). Precision agriculture (PA), which seeks to apply the right amount of inputs (fertilizers, irrigation water, pesticides, and other chemicals) in the right location at the right time, is essential to meet the requirements of future global food production, as well as environmental sustainability and climate resilience. PA is predicated on accurate sensor measurements, timely and sound decision-making, and automated actuators. The backbone of PA is the Internet of Things (IoT) technology that automates data collection, data analytics, data presentation, control, and efficient data communication (Chamara et al., 2022).

Imaging sensors or digital cameras are essential for PA as they can capture more information than traditional scalar or vector sensors.

* Corresponding author at: Department of Biological Systems Engineering, 203 L.W. Chase Hall, East Campus, University of Nebraska-Lincoln, Lincoln, NE 68583, USA.
E-mail address: yge2@unl.edu (Y. Ge).

https://doi.org/10.1016/j.compag.2023.108420
Received 23 April 2023; Received in revised form 8 October 2023; Accepted 6 November 2023
Available online 16 November 2023

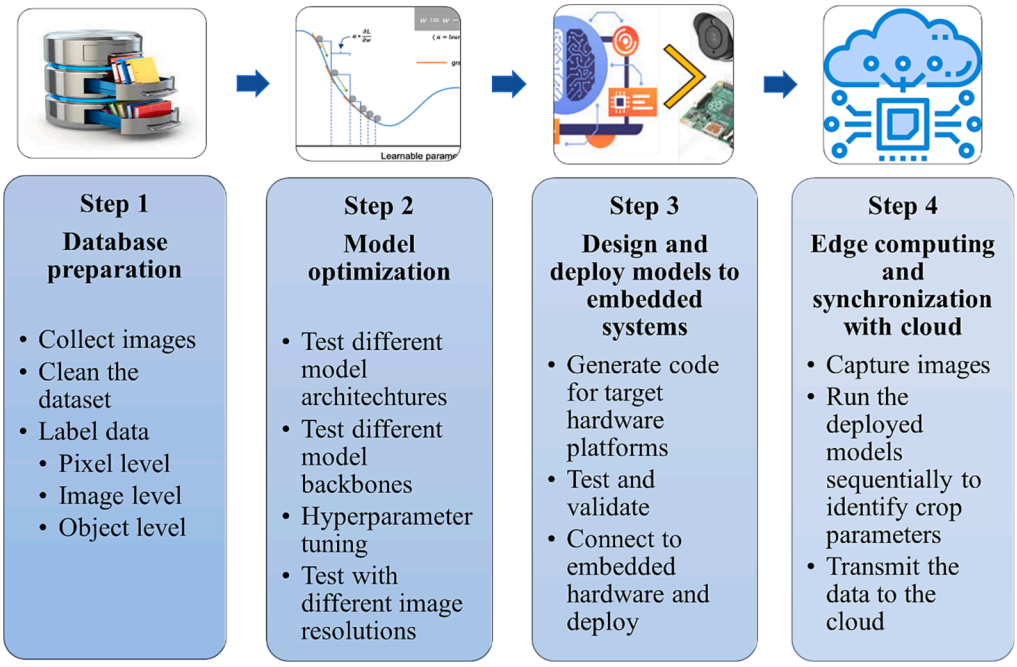


Fig. 1. Steps of edge image processing program deployment on the embedded system (edge devices).

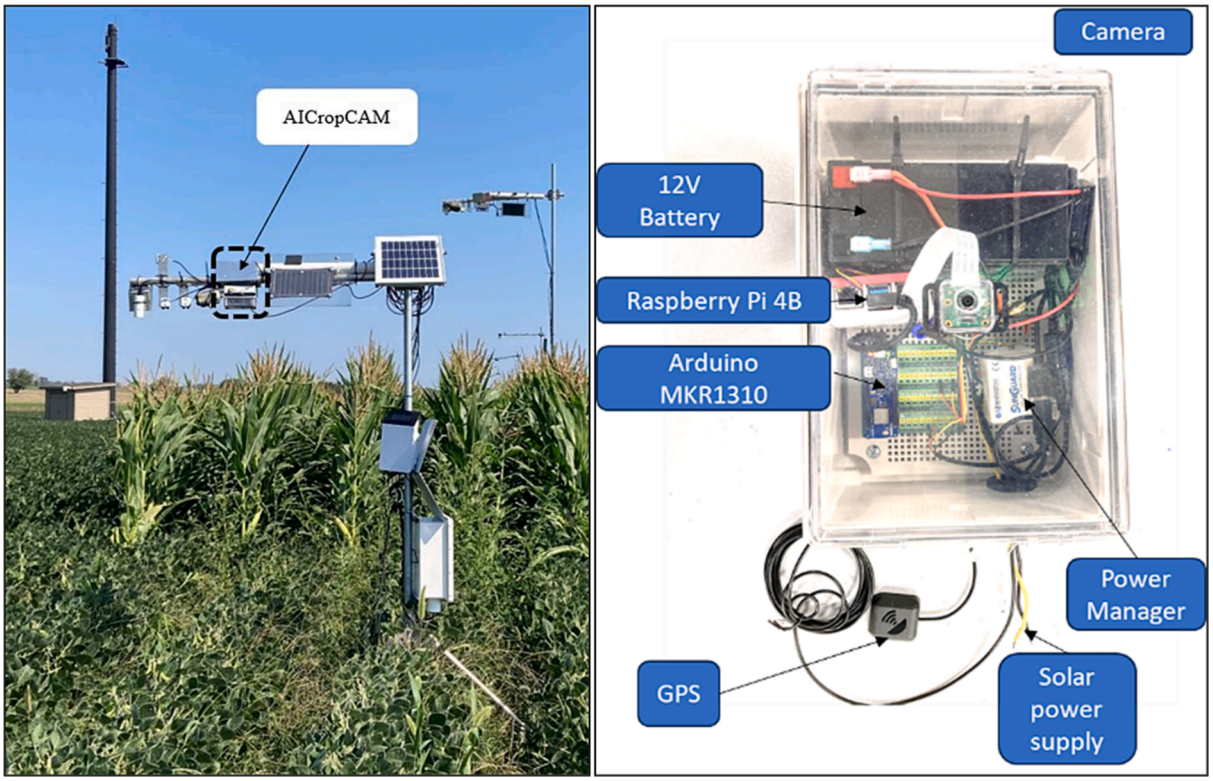


Fig. 2. Left: An Illustration of how AICropCAM was set up in the field for image collection. In addition to the camera, other components such as the solar panel and data logger were also shown. Right: A close-up view of AICropCAM and its hardware components.

Images can capture crop phenology for precise decision-making (Taylor and Browning, 2022, Tian et al., 2020). Cyclic events such as vegetative growth, flowering, leaf count and color change, maturation, and senescence are studied in crop phenology, which is essential to PA as it determines the management inputs required by crops. Moreover, images have rich information on the scene that allows for pest pressure evaluation. At present, a limited number of sensors are available for pest identification and pest pressure estimation. Among them, imaging sensors provide the most promising solution.

Conventional (handcrafted feature extraction) and Artificial Intelligence (AI)-based image processing are the two branches of image processing. Traditional approaches extract image features defined by shape, texture, and color (Anubha et al., 2019; Yuan et al., 2019). The AI-based methods use Convolutional Neural Networks (CNN) to extract features

Table 1
Annotation criteria used to generate labels from the images to train and test the four deep convolutional neural network models in AICropCAM.

Labeling Type	Class	Description
Image classification (CropClassiNet)	Rejected	Images were labeled as rejected due to multiple reasons: blurred images caused by water droplets on the lens; the cameras turned away from the targeted crop; crops growing up to the camera blocking the view or capturing only a few leaves; people present in the images; lens covered with different stuff; and the camera was not installed in the field.
	Corn	Images entirely covered by corn plants at different growth stages.
	Soybean	Images entirely covered by soybean plants at different growth stages.
	Grass/ Weed	Images only comprise grass/weed plants at different growth stages.
	Night	Images captured under low lighting conditions. Most of the cameras were not programmed to stop collecting images under low light.
Crop canopy and background (CanopySegNet)	Canopy	Pixel labeling was done on the crop canopy. We used assisted freehand tool and the superpixel option in the MATLAB image labeler.
	Background	Pixel labeling was done on the crop canopy. We used assisted freehand tool and the superpixel option in the MATLAB image labeler.
Plant-type (PlantCountNet)	Weed	Weed present in the image was labeled using bounding boxes. It was challenging to locate the weed after the corn or soybean canopy was closed.
	Soybean	Soybean plants present in the image were labeled using bounding boxes.
Insects (InsectNet)	Insects	During the labeling process, without distinguishing insects based on their type, all the insects present in the images were labeled using bounding boxes.

from images (Luis et al., 2020). CNN models with multiple hidden layers for feature extraction and learning are considered Deep Convolutional Neural Networks (DCNN) (LeCun et al., 1998).

Conventional imaging platforms in PA store images locally using onboard storage memories. Post processing refers to the processing of images stored at the central data storage in batches at a later time to extract useful information (Aasen et al., 2020). Imaging platforms that can access the internet through a stable connection with high bandwidth can automatically upload images to Cloud data storage. The vast majority of farmlands worldwide are in rural and remote areas with poor access to electric power and internet connectivity. This represents a big challenge for camera systems deployed in rural farmlands for high-speed image processing, data transmission, and low-latency decision-making (Richardson, 2019). Post-processing of crop images has been used for the estimation of leaf area index (Aasen et al., 2020), growth rate (Sakamoto et al., 2012), leaf chlorophyll and nitrogen content (Wang et al., 2014), fruit counts (Wang et al., 2014), and plant height (Sritarapipat et al., 2014). Further post-image processing allows for the assessment of biotic stress, such as pest density (Barbedo, 2014; Park et al., 2007) and weed pressure (Wang et al., 2019), as well as abiotic stress, such as nutrient deficiency (Ghorai et al., 2021).

Richardson (2019) suggested that deep learning-based methods have the potential to facilitate the extraction of more sophisticated phenological data from both new and previously archived camera imagery compared to conventional image processing. Semantic segmentation-based canopy coverage (CC) estimation (Chamara et al., 2021; Liang et al., 2023), image classification-based crop identification (Anubha et al., 2019), disease identification (Sharma et al., 2020), growth stage prediction (Yasrab et al., 2021) and object detection-based plant feature identification (A. Wang et al., 2019) are examples of DCNN applications in agriculture. Conventional image processing requires less computational power and less energy, but they are limited in adaption to new scenarios, while deep learning requires high computational power and consumes more energy. DCNN models require large memory due to the large number of parameters these models hold. Therefore, it is not easy to implement these models practically in embedded systems that have less memory and computation power. These models also require a large amount of data to train to predict with high accuracy. Therefore, it is resource intensive.

Edge image processing is the image processing done on image-capturing devices. The main advantage of edge image computing is that it lowers the high throughput data transmission requirement over a wireless IoT-enabled imaging network (Cao et al., 2020). Wang et al. (2022a) demonstrated the capability of identifying potted flowers with precision above 89 % in real-time in a Jetson TX 2 computing module based on a DCNN algorithm. These authors suggested that a cloud-edge collaborative framework could achieve real-time and automatic learning for the DCNN model they have developed. Wang et al. (2022b) proposed a real-time weed detection model run on Jetson AGX Xavier for field robots. The authors proved it was possible to do real-time weed detection with a precision above 90 % yet required expensive hardware. Wang et al. (2022a) reviewed Raspberry Pi single-board computer-based real-time image processing applications. They concluded that Raspberry Pi (Datasheet Raspberry Pi Model B, 2019) is a cost-effective edge computing unit that could potentially be used as an edge image processing unit, and the capability of integrating it with IoT was also discussed. Zualkernan et al. (2022) demonstrated an edge image processing platform for the classification of animals and transmitting the identified animal and time of identification via LoRa for a camera trap.

Past literature on IoT and image processing applications in agriculture has highlighted a research gap in edge image processing with IoT-enabled crop monitoring cameras. In-field crop cameras are expected to make real-time crop management decisions based on real-time image processing; however, poor internet connectivity in agricultural fields severely limits their capability. To address this gap, we have developed a novel imaging platform named AICropCAM that extracts plant and crop canopy level parameters through DCNN and uploads them to the Cloud via low-power, low-throughput communication protocols. We also demonstrated AICropCAM on an IoT-enable wireless sensor network in corn and soybean fields.

A technology that addresses image processing at the lowest level (edge) and transmits only useful information can revolutionize real-time

Table 2
DCNN model image allocation and image augmentation.

Model	Number of images				Data Augmentation Techniques
	Total	Training	Validation	Test	
CropClassiNet	43,611	30,528	9,810	3,273	Random rotation, random X and Y reflection
CanopySegNet	51	31	10	10	Transformation (random left/right reflection and random X/Y translation of ±10 pixels)
PlantCountNet	110	88	11	11	Transformation (same as CanopySegNet)
InsectNet	542	326	108	108	Transformation (same as CanopySegNet)

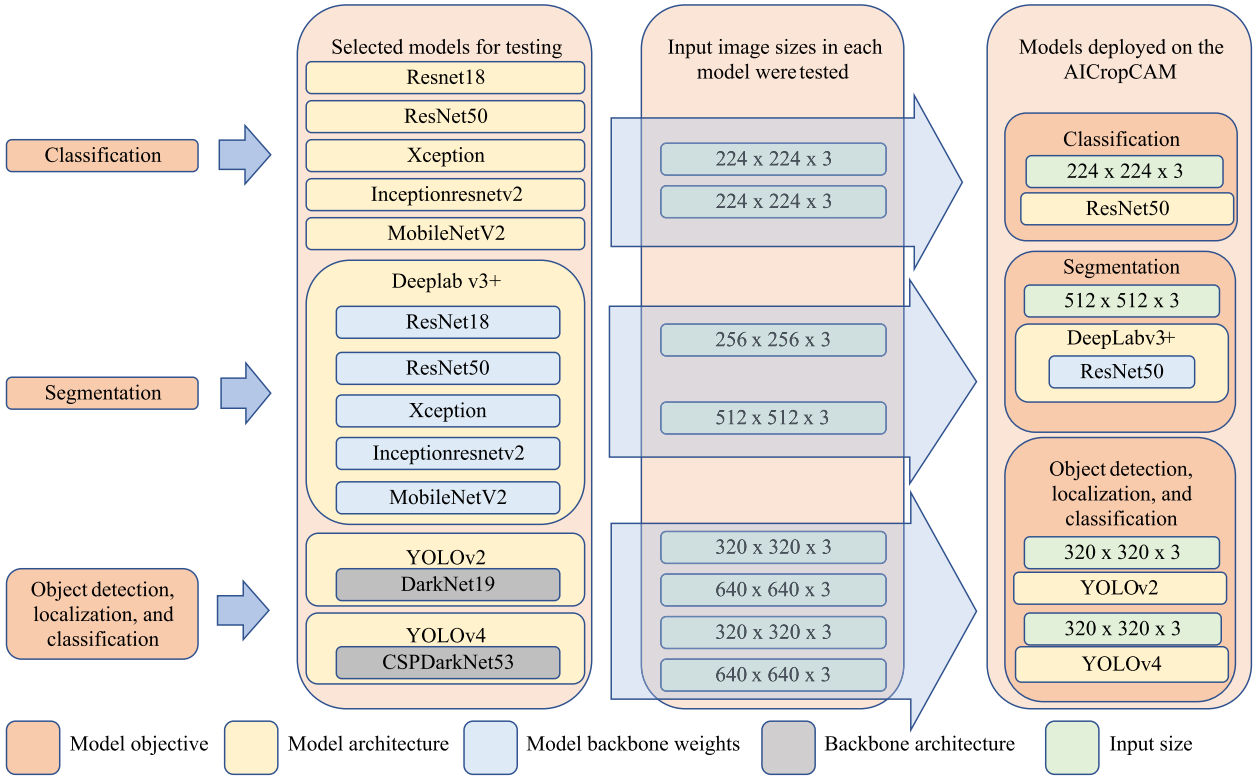


Fig. 3. DCNN model selection process during the training and testing by attempting different model architectures, model backdone weights, and input image sizes.

Table 3
Hyperparameter values and training options for the best models (SGDM - stochastic gradient descent with momentum, RMSProp - Root mean square propagation).

Training option and the function/Hyperparameters	Values for CropClassiNet	Values for CanopySegNet	Values for InsectNet (320 × 320 × 3)	Values for PlantCountNet (320 × 320 × 3)
Optimizer	SGDM	SGDM	SGDM	RMSProp
Momentum	0.9	0.9	0.99	NA
Initial learning rate	0.001	0.001	0.001	0.001
Learn rate schedule	Piecewise	Piecewise	Piecewise	Piecewise
Learn rate drop period	10	10	10	10
Learn rate drop factor	0.3	0.3	0.1	0.3
Minibatch size	16	4	16	32
L2Regularization	NA	0.005	0.005	0.005
Validation frequency	3	3	3	10
Shuffle	Every epoch	Every epoch	Every epoch	Every epoch
Validation patience	4	10	10	10
Max epochs	100	300	1000	100
Execution environment	Multi GPU	Multi GPU	GPU	GPU

decision-making in PA. Therefore, the main objective of this paper is to demonstrate AICropCAM to perform edge image processing and low-throughput, low-power, and long-range data transmission through IoT technology. In this novel AICropCAM platform, multiple DCNN image processing algorithms run in series to extract plant-level and canopy-level features in an embedded system. Image classification, object detection with classification, and image segmentation are the three major applications of DCNN image processing, and all three are included in AICropCAM to demonstrate the capabilities of DCNN for image processing in PA. AICropCAM has trained models for canopy segmentation, crop classification, plant growth stage identification, plant counting, weed counting, and plant type identification. All the protocols that transmit data from AICropCAM to the Cloud were custom designed. AICropCAM sends the generated data to a cloud platform for logging, visualization, and analysis. Furthermore, this paper explains the DCNN model training process, model performance, and test results. We reported the model training comprehensively because it was essential for AICropCAM development.

2. Materials and methods

Essential activities in this research were data/image collection and preprocessing, hardware design for AICropCAM, software design for data transmission between the edge and the Cloud, deep learning model design, and model training and optimization (Fig. 1). AICropCAM was implemented in a corn and soybean field at the field phenotyping facility in Mead, Nebraska, USA (Bai et al., 2019). We demonstrated the training of the following DCNNs: CropClassiNet for classifying images based on image quality and crop type, CanopySegNet for segmenting crop canopy from the background, PlantCountNet for classifying and counting soybean and weed plants, and InsectNet for identifying insects and counting them.

2.1. Image collection, annotation, preprocessing, and augmentation

Image collection for DCNN model training occurred in four growing seasons using three different types of cameras: (i) commercially

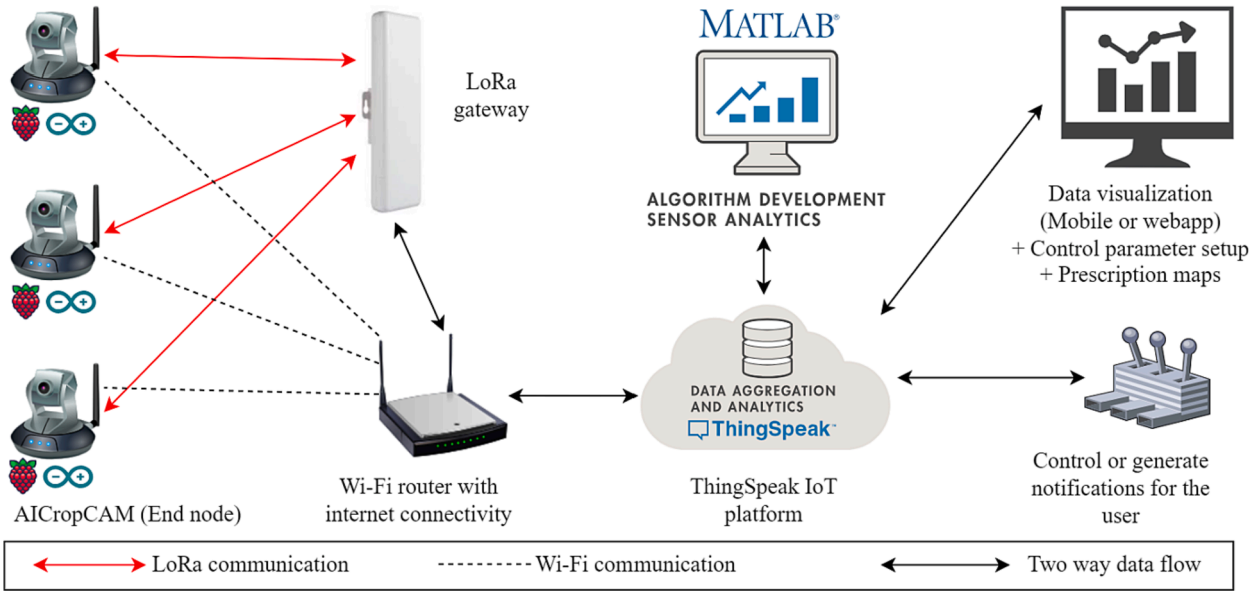


Fig. 4. Hardware overview of AICropCAM and data flow.

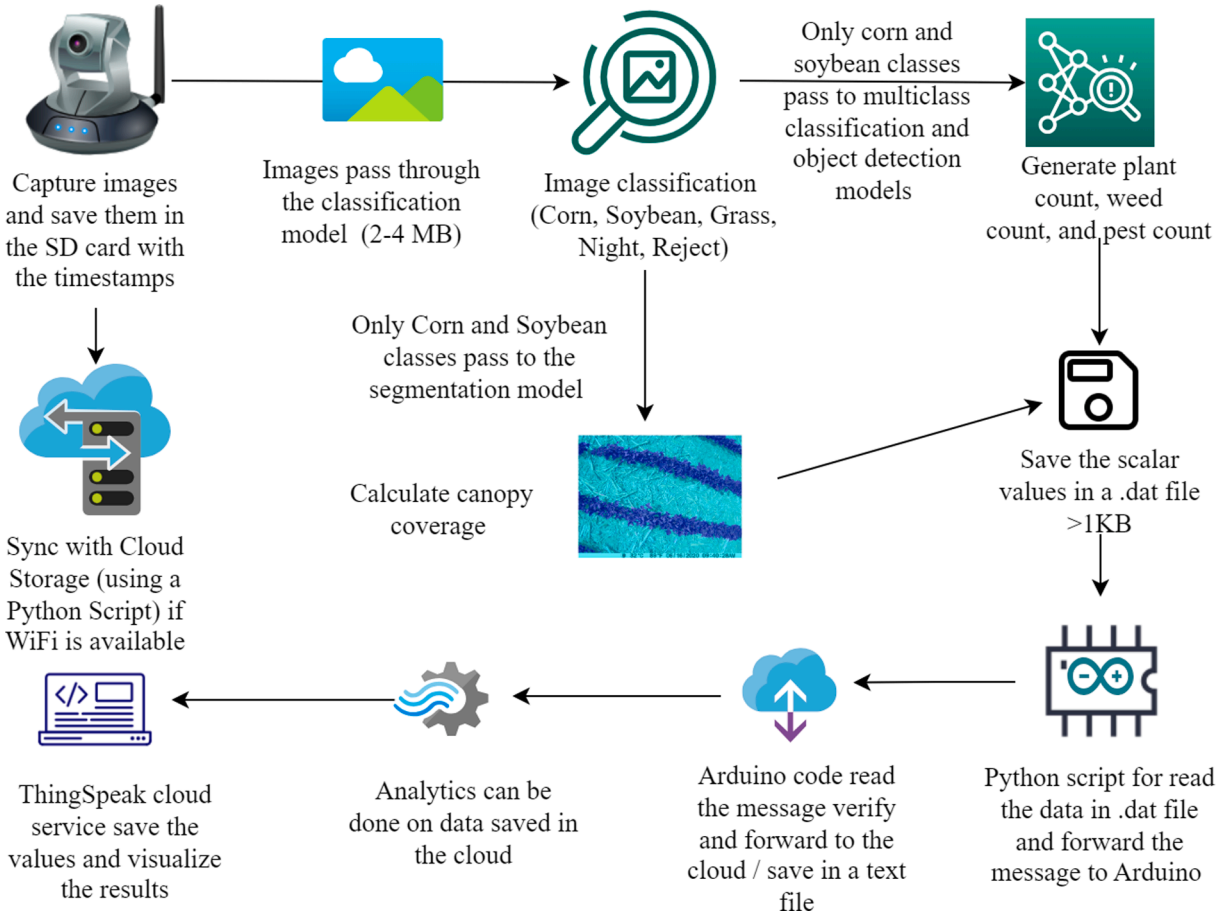


Fig. 5. Overall sequential image processing and data generation flow chart.

available Meidas SL122 trail cameras in 2019 (Meidas Trail Cameras, 2022), (ii) OV5642 imaging sensors with ArduCAM camera shields in 2020, and (iii) Raspberry Pi Camera Module V2 with Raspberry Pi Zero in 2021 and 2022 (Chamara, 2021). All the cameras were mounted on the bars horizontally extended and fixed on stationary poles erected vertically in the fields, as shown in Fig. 2A. The distance between the crop canopy and the cameras was maintained between 0.5 and 1.5 m throughout the growing seasons. Images used for training the InsectNet were also captured with smartphones as we could not collect enough images with insects from the three types of cameras mentioned above.

Table 4
List of parameters used to represent information in the images.

Parameter	Abbreviation	Represent information
Image location	LOC	Node ID manually entered/Global positioning system location coordinates
Image orientation	IO	Accelerometer/Manually feed/Gravity switch
Image quality/ Crop type	CT	Image classification based on image quality and the crop type
Plant count/Weed count	PC/WC	Multiclass object detection/classification
Crop canopy coverage	CC	Semantic segmentation
Pest count	PstC	Multiclass object detection/classification

All three standard image annotation techniques in deep learning model training were utilized: (1) folder labeling for the image classification models, (2) pixel-level annotation for the semantic segmentation model, and (3) bounding boxes for object detection models. Images belonging to the same class were grouped into a single folder, and five distinct classes (or folders) were created: rejected, corn, soybean, grass, and night. Separating the crop canopy from the soil was done with pixel-level annotation and semantic segmentation. Bounding boxes, the smallest rectangle around an object, were drawn for corn plants, soybean plants, weed plants, and insects. Table 1 explains each type of annotation used in the model training.

Image preprocessing is necessary for DCNN model training and real-time edge image processing. Differences in the input layer size in different DCNN models demand that images be resized before passing through the model for training or prediction purposes. High-resolution images improve accuracy but require more computational power. For specific applications, labeled datasets were only limitedly available. Therefore, image augmentation techniques were used to increase the number of image data sets, including scaling, flipping, cropping, rotation, color transformation, PCA color augmentation, and noise rejection (Paymode and Malode, 2022). Multiple augmentation techniques were used for each model, as detailed in Table 2. Additionally, Table 2 provides the numbers of images in training, validation, and testing for the four DCNN models.

Our main objective was not to make the most accurate prediction for the DCNN models but to demonstrate the concept of implementing edge image processing and transmitting the results to the Cloud for decision-making. Therefore, we selected a limited number of images for CanopySegNet, PlantCountNet, and InsectNet, which were sufficient to train models with a reasonable degree of accuracy.

2.2. DCNN model architecture selection, training, evaluation, and deployment on the edge device

The steps to select model architecture/model backbone weights and image input sizes to train the best model for CropClassiNet, CanopySegNet, PlantCountNet, and InsectNet are summarized in Fig. 3. Unlike many DCNN applications that prioritize higher accuracy, our application focused on finding the balance between accuracy and model deployability on the edge device.

For example, in the development of CropSegNet (Segmentation), we selected DeepLabv3+ (Firdaus-Nawi et al., 2018) with weights initialized from pre-trained networks of ResNet18 (He et al., 2016), ResNet50, Xception, InceptionresnetV2, and MobileNetV2. The input image sizes tested were $512 \times 512 \times 3$ and $256 \times 256 \times 3$, and training options were kept constant to find the best-performing networks, which should also be deployable to Raspberry Pi 4B. This process identified DeepLabv3 + with ResNet50 as the most suitable model for CropSegNet, with an input image size of $512 \times 512 \times 3$. Table 3 summarizes the hyperparameter values and training options for the final DCNN models deployed to the edge device.

$$Accuracy = \frac{Number\ of\ accurate\ predictions}{Total\ images\ in\ test\ dataset} \times 100\% \quad (1)$$

$$Precision = \frac{True\ positive}{True\ positive + False\ positive} \quad (2)$$

$$Recall = \frac{True\ positive}{True\ positive + False\ negative} \quad (3)$$

$$F1\ Score = \frac{2 \times precision \times recall}{(recall + precision)} \quad (4)$$

$$\text{Jaccard index} = \frac{|Target \cap Prediction|}{|Target \cup Prediction|} \quad (5)$$

$$Intersection\ over\ union(IoU) = \frac{Intersection\ area}{Union\ area} \quad (6)$$

$$Mean\ Average\ Precision = \frac{1}{n}\sum_{k=1}^{k=n} AP_k \quad (7)$$

The performance of the four DCNN models was evaluated using the indices calculated from Eq. (1) to (7). Accuracy, Precision, Recall, F1 score, and Jaccard index were used for the classification models CropClassiNet and CropSegNet, whereas IoU and mAP (Mean Average Precision) were used for PlantCountNet and InsectNet. Jaccard index gives the proportion of correctly predicted labels to the total number of labels. Model training was performed on an NVIDIA GeForce GTX 1650 Ti Mobile processor, a dedicated mid-range graphics card with 4 GB GDDR6 memory on a Dell XPS 15 9500 Laptop. The laptop had an Intel Core i7-10750H 10th Gen processor,16 GB DDR4 RAM, and 1 TB SSD hard disk.

2.3. Hardware and software of AICropCAM

The IoT data transmission and edge image processing hardware comprised the following major components: a Raspberry Pi 4B single-board computer, an Arduino MKR1310 development board, an Arduino MKR Relay Proto Shield, and a Dragino OLG02 outdoor dual channels LoRa Gateway (Fig. 4). The 12 V 8Ah battery powered the Raspberry Pi 4B, controlled through the relay shield managed by the Arduino MKR1310. A 3.7 V lithium polymer battery powered the Arduino MKR1310 board. There are two advantages of having a separate Arduino board. First, the Arduino board consumes less power than the Raspberry Pi 4B module. It can be switched on and off according to user requirements. Second, it allows uninterrupted communication between the edge node and the Cloud with low power.

AICropCAM required programming on two hardware platforms. Arduino was programmed using C++ in Arduino's Integrated Development Environment. Raspberry Pi imaging and image processing program was developed in MATLAB and deployed onto the Raspberry Pi 4B using the MATLAB Coder and MATLAB Compiler. A python program was designed to read the saved data in the Raspberry Pi 4B and serially communicate to the Arduino MKR1310. The primary functions of the MRK1310 program were to (1) turn on the Raspberry Pi 4B module based on the user-defined time intervals, (2) get the processed data, including the results of DCNN model predictions, through serial communication from the Raspberry Pi 4B, and (3) transmit the data to the ThingSpeak Cloud channel through the LoRa gateway. All the DCNN models were trained using the MATLAB deep learning toolbox. In the edge deployment, a MATLAB program runs multiple models logically depending on the prediction result of the previous model estimation, as shown in Fig. 5. MATLAB coder generated the C and C++ code derived from the program we developed to run on the Raspberry Pi. MATLAB Compiler generated the standalone application on the Raspberry Pi (The MathWorks, 2022).

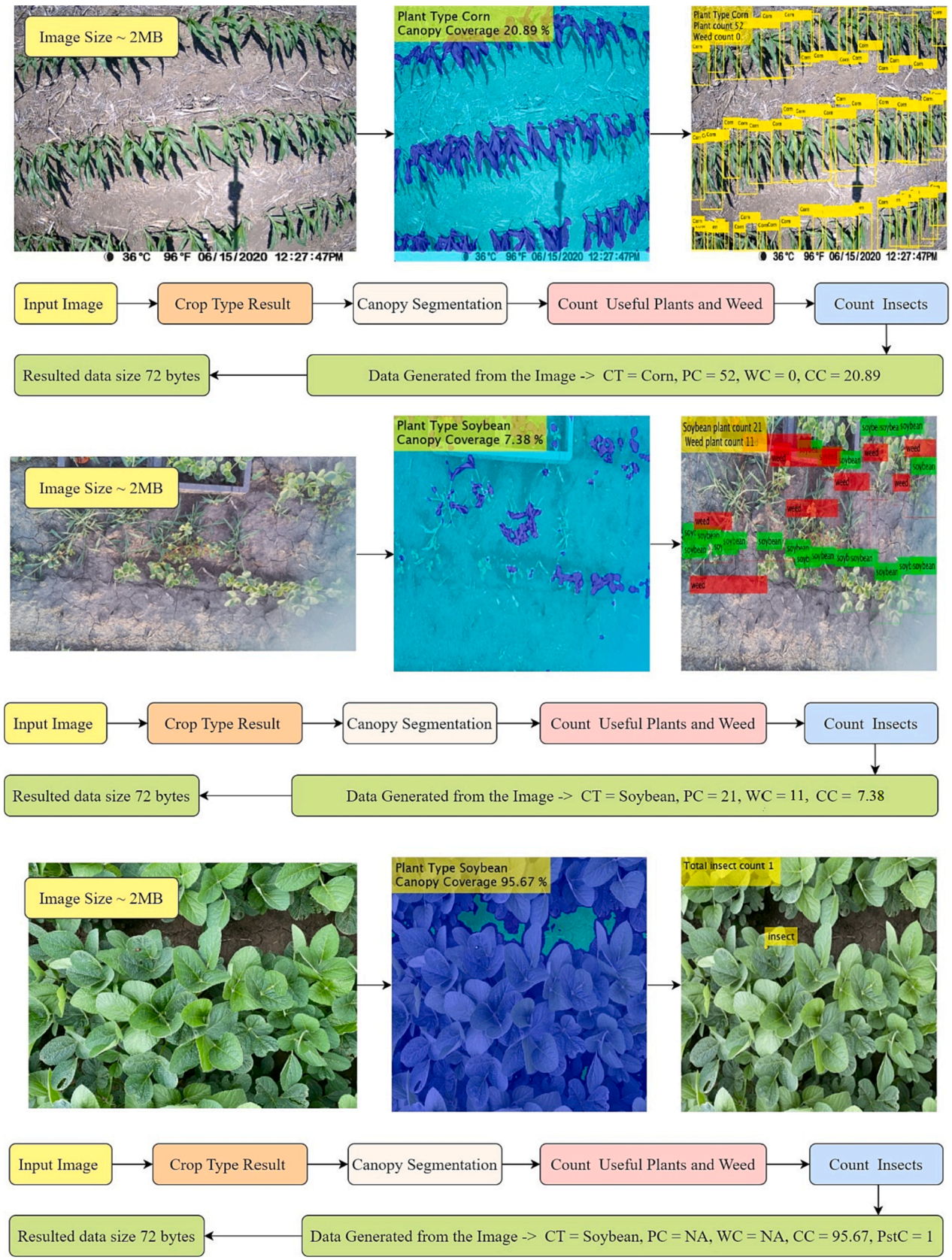


Fig. 6. Examples of message generation and data size reduction for LoRa transmission.

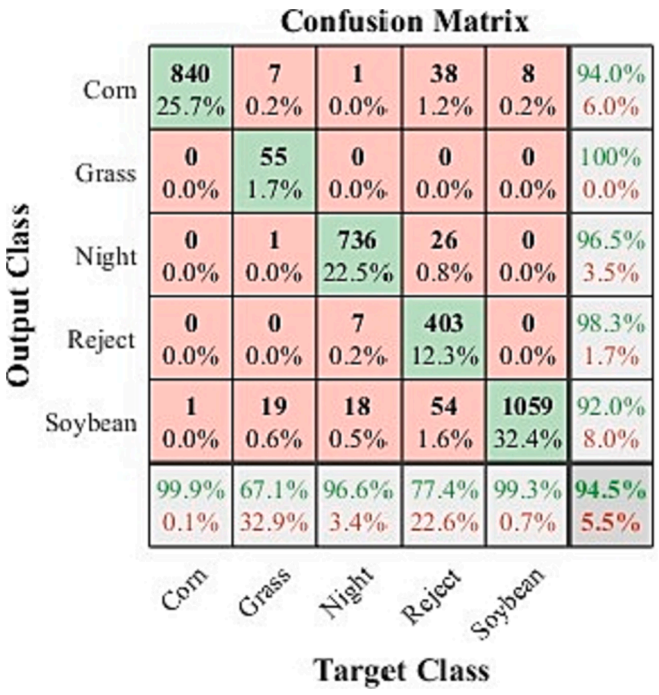


Fig. 7. Confusion matrix for test images by CropClassiNet.

Table 4 lists the parameters generated by the models in AICropCAM. The abbreviations in Table 4 are fields holding data in the program to reduce the complexity of system development and maintain a common standard among different platforms. Fig. 6 shows the data generation from images. According to Fig. 6, the size of the images were around 2 MB before being fed into the image processing pipeline. The output message contains the crop type (CT), plant count (PC), weed count (WC), canopy coverage (CC), and pest count (PstC). The resulting message is typically less than 100 bytes. This represents a substantial reduction of memory size with the output being 0.00005 times the size of the original image. Consequently, this message can be transmitted in a single message via LoRa as the maximum LoRa packet size is around 256 bytes.

2.4. Data transmission, visualization, and storage

The data generated after image processing were saved on the Raspberry Pi 4B SD card, allowing access to the data remotely or through manual retrieval during field visits. Two options for transmitting the collected data to the ThingSpeak IoT platform are available. Firstly, the data can be uploaded directly from the Raspberry Pi 4B if internet connectivity is available for growers with Wi-Fi access. Secondly, the Raspberry Pi 4B transmits the recently acquired data to the Arduino MKR1310. The Arduino MKR1310 decodes the data received from the Raspberry Pi 4B and forwards it to the ThingSpeak. The second method is for low-rate, long-range communication beyond the limit of Wi-Fi.

A single message receivable to the ThingSpeak server includes data for eight fields. In our demonstration, a single message was enough to transmit the data generated. Fields 1 and 2 are reserved for geographic coordinates (namely, latitude and longitude) to represent the device's location. The third field was for camera orientation. Image quality/crop type, plant count, weed count, insect count, and crop canopy coverage were allocated from fields four to eight. ThingSpeak supports eight channels per gateway. If additional data is generated in the future, we have to create new channels to accommodate them. However, only data in a single channel can be passed through a single message. The Arduino-LoRa library was used to prepare the LoRa messages forwarded to the gateway (Mistry, 2016). The message generated from the Arduino MKR1310 includes the device identification number and the data with the field number. Once the gateway receives this message, it adds the target client ID (generated by ThingSpeak when defining a device), host address (mqtt://mqtt3.thingspeak.com), server port number, username and password, channel ID, and the data in each field according to the Message Queuing Telemetry Transport (MQTT) protocol. Username and password ensure that only authorized devices can transmit data to the ThingSpeak platform.

ThingSpeak provides two ways to interact with its platform, REST (Representative State Transfer) and MQTT protocols. The advantages of using MQTT over REST protocol are that it supports ThingSpeak data publishing, including immediate and minimum power consumption and data transmission over limited bandwidth, which encouraged us to select the MQTT protocol in our demonstration.

3. Results and discussion

3.1. DCNN model performance

CropClassiNet had a test accuracy of 91.26 %, a Jaccard Index of 0.77, and an F1-score of 0.91; the confusion matrix is given in Fig. 7. The highest precision is for the "grass" class (100 %), and the lowest is for "soybean" (92.0 %). The highest recall is for the "corn" class (99.9 %), and the lowest is for "grass" (67.1 %). The primary goal of CropClassiNet

Fig. 8. An image of soybean crop and the segmentation result by CropSegNet to calculate canopy coverage.

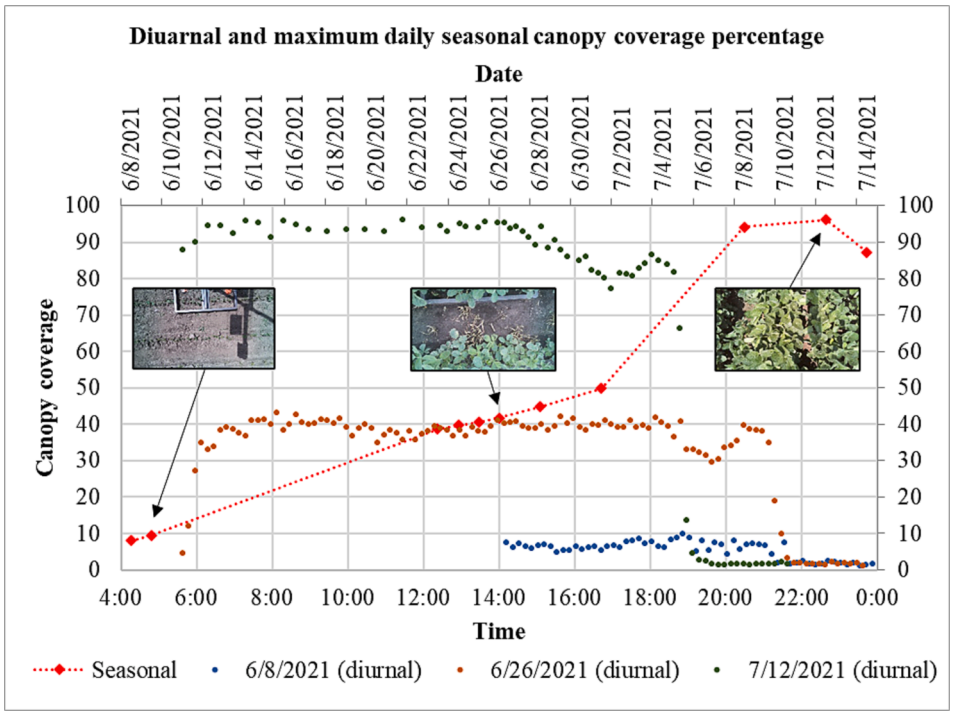


Fig. 9. Examples of diurnal and seasonal variations of canopy coverage as computed by CropSegNet.

Table 5
Performance of PlantCountNet and InsectNet on the test image set (Root mean square error (RMSE)/Final validation loss (FVL)).

Model Name	Architecture	Input size	Validation RMSE/FVL	Mean average precision	Object class
PlantCountNet	YOLOv2	320 × 320 × 3	0.888 (RMSE)	0.66 0.86	Soybean Weed
InsectNet	YOLOv4	320 × 320 × 3	26.2 (FVL)	0.02	Insect

is to determine the quality of new images and direct them for subsequent processing (Fig. 5). This step has never been executed in an image-based crop monitoring platform before. Further, CropClassiNet can eliminate erroneous images when humans are present in the camera's field of view or when the camera is misaligned due to external factors. AICropCAM can send maintenance requests through IoT analytics if rejected images are continuously generated.

CanopySegNet on the test images achieved a global accuracy of 0.93, a weighted IoU of 0.87, and a mean BF score of 0.73. Fig. 8 shows an example of an original soybean image and the corresponding segmentation result by CanopySegNet, which estimated CC to be 18.72 %. Season-long, time-series images can be fed into CanopySegNet to generate diurnal and seasonal curves of crop CC, as shown in Fig. 9.

According to Fig. 9, canopy coverage percentage variation is low during the daytime and reaches zero at night. This verifies the need to eliminate low-light images before segmenting. As shown in Fig. 5, it is possible to eliminate the generation of false values when the camera captures images under low light conditions by halting the process of running CanopySegNet. There are three diurnal variation series on 6/8/2021, 6/26/2021, and 7/12/2021 in Fig. 9. The CC increased from 8 % to 95 % between 6/8/2021 to 7/12/2021. The seasonal trend showed that the CC reached a maximum around 7/8/2021. These results suggest that the proposed stacked models can track the daily and seasonal CC variation and eliminate the effect of lighting conditions on false value generation.

The overall performance of the PlantCountNet and InsectNet is given in Table 5. Fig. 10(A) and 10(B) show the result obtained by PlantCountNet for a soybean image at an early vegetative stage (V3). Meanwhile Fig. 10(C) and 10(D) shows the result at a reproductive stage (R1). It can be seen that, at V3 stage, the model outputs matched the labels of soybean and weed plants well, indicating a level of high accuracy.

The size of insects is very small compared to the size of images (Fig. 11), which is the main reason for the low mAP for InsectNet (Table 5). Increasing input image resolution beyond 480 × 480 × 3 is impractical as it exceeds the memory limitation to load models into Raspberry Pi 4B. A potential solution could be to increase the resolution of the region of interest by splitting the original image while keeping the resolution the same. Also, we suggest using the approach recommended by Tetila et al. (2020) in the future on Raspberry Pi model 4B. As technology advances, we expect the memory capacities will increase for edge computing units. At the same time, the state-of-the-art object detection algorithms will improve the accuracy for small object detection.

3.2. Power consumption for Raspberry Pi 4B

Since edge cameras in farmlands have limited access to electric power, information on their power consumption is essential for designing IoT devices and systems. AICropCAM is designed to be energized by solar power. It runs on a rechargeable battery when there is no solar power. We monitored the maximum energy consumption of each task performed by AICropCAM, and the result is presented in Table 6. Four main strategies are available for the power management of IoT edge devices: Selecting power-efficient hardware, maintaining low power modes, dynamic power management, and cloud-based management. Raspberry Pi 4B is an affordable power-efficient single-board computer suitable for our application, but it does not naturally support low-power modes. Therefore, we introduced the Arduino MKR1310

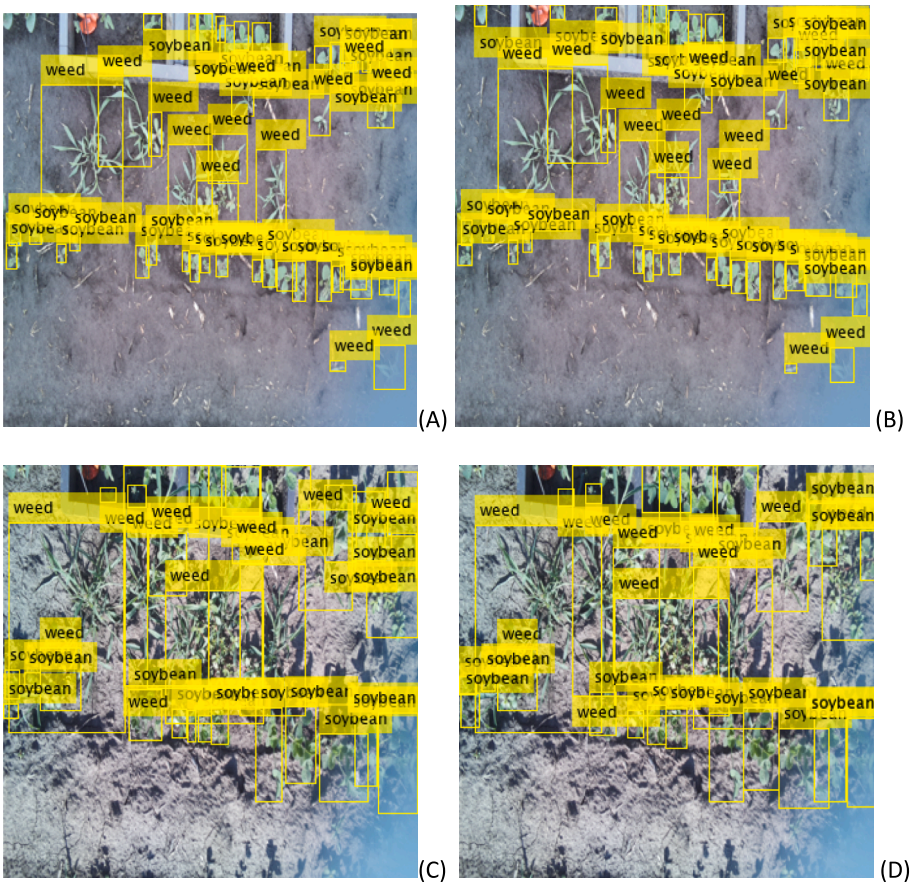


Fig. 10. The result of PlantCountNet for soybean and weed counting: Manually annotated vs. model-predicted bounding boxes at V3 growth stage (A and B); manually annotated vs. model-predicted bounding boxes at R1 growth stage (C and D).

LoRa module for the Raspberry Pi 4B dynamic power management. Furthermore, this Arduino module allows us to perform cloud-based central management independently.

For our measurements, we used a Raspberry Pi 4B with 8 GB of RAM, connected to an HDMI monitor, a USB keyboard, and a USB mouse, and ran a MathWorks® Raspbian image (file used to boot the Raspberry Pi 4B). The Raspberry Pi 4B was operated at room temperature and connected to a wireless LAN access point and a laptop via an Ethernet cable. The electric current consumption for running each DCNN model was recorded during the test. CropClassiNet had the highest current consumption, while the PlantCountNet and InsectNet models had the lowest. As for LoRa transmission, we could not measure its current consumption because the lowest value our instrument could measure was 0.01A. Based on the manufacturer's specifications, the Arduino MKR1310 consumes 104 uA at 5 V.

The average time to run the DCNN models is essential to estimate the energy consumed for each prediction. These parameters listed in Table 7 provide essential guidelines for designing IoT sensor nodes with suitable batteries and power sources. We also noticed that typically the first prediction of a model took the longest time, but the rest take a considerably shorter time to predict.

Semantic segmentation was the most power-demanding activity, while insect detection was the least. Changing the order of the image processing models and adding new models or dropping existing models is possible during regular operation. It enables dynamic power management within the Raspberry Pi module.

The main advantage of AICropCAM is that it implements a stack of four DCNN-based image processing models with multiple objectives. To the best of our knowledge, this is the first time such a system has been developed for a field crop monitoring camera. AICropCAM has applications such as setting up smart in-field or greenhouse IoT camera networks with edge computing capability, monitoring crops by attaching them to sprinkler irrigation systems (pivots and linear moves), or collecting crop information through ground or aerial mobile robots. The relatively short time to run each DCNN model makes the system suitable for real-time applications, including variable rate irrigation, fertilization, and spraying. For example, a pivot irrigated multi-cropping system with AICropCAM can automate irrigation or fertigation transition between different crops or crops at different growth stages by automatically providing the crop type or growth stage information to the irrigation controller. Additionally, existing herbicide or pesticide sprayers can get the feedback of the PlantCountNet and InsectNet in the AICropCAM for precision spraying.

4. Conclusion and future perspectives

This paper outlines the essential components of constructing a functional edge image processing framework for real-time crop monitoring. From a software standpoint, CropClassiNet can categorize captured images according to image quality and detect the presence of specific crop types for further processing. CanopySegNet can further quantify the degree of canopy coverage; PlantCountNet can count the number of plants and weeds in the image; and finally, InsectNet can count the number of insects in the image. These four DCNN models, when implemented on edge devices, can extract an array of important crop and canopy parameters from field images and enable real-time, low-latency decision making and applications.

Deep learning-based image processing on the edge has excellent potential in PA. Applications of AICropCAM are not limited to image classification, segmentation, plant counting, or weed counting. Potential

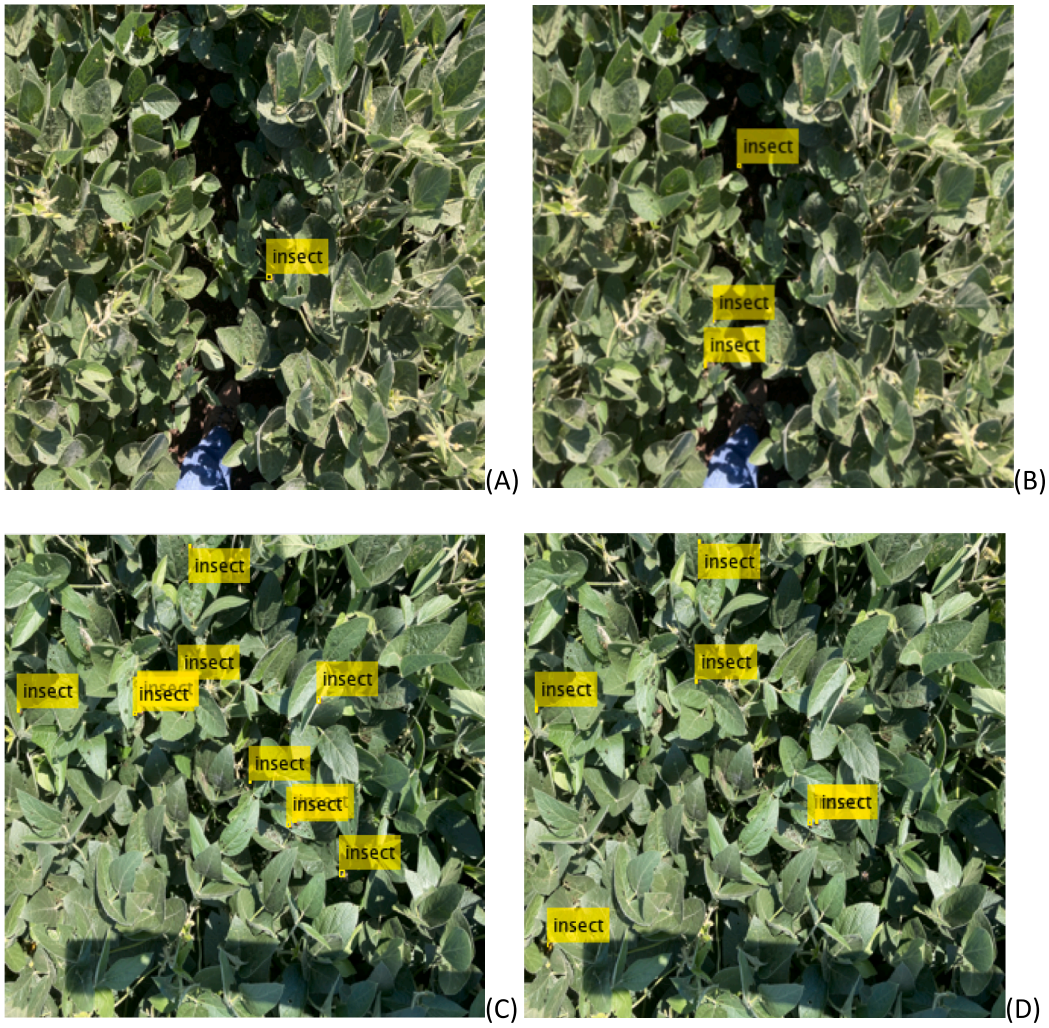


Fig. 11. The result of InsectNet for insect counting in soybean. The top row shows a situation of high false positives and low false negatives: (A) and (B) are manually annotated and model-predicted insect labels, respectively. The bottom row shows a situation of low false positive and high false negative: (C) and (D) are manually annotated and model-predicted insect labels.

Table 6
Electrical power consumption of the Raspberry Pi 4B and the Arduino MKR1310 during edge image processing.

Device	Activity	The maximum current range and the voltage recorded
Raspberry Pi 4B	Idle run	5.25 V × (0.45 – 0.53) A
	Image classification	5.25 V × (0.97 – 1.04) A
	Image segmentation	5.25 V × (0.98 – 1.11) A
	Weed and plant detection	5.25 V × (0.62 – 0.70) A
	Insect detection	5.25 V × (0.62 – 0.70) A
Arduino MKR1310	Sleep mode	<0.01A
	Serial communication	<0.01A
	LoRa transmission	<0.01A

Table 7
Time duration needed for the selected DCNN models deployed in the Raspberry Pi 4B.

Model/Task	Input image size	Time for predicting results (s)	The maximum power demand for the activity (W)
CropClassiNet/Image quality evaluation and crop classification	224 × 224 × 3	6.44	5.46
CanopySegNet/Semantic segmentation to separate canopy from background	512 × 512 × 3	20.20	5.83
PlantCountNet/Weed and plant detection, classification, and counting	320 × 320 × 3	14.38	3.68
InsectNet/Insect detection	320 × 320 × 3	0.20	3.68

future applications include insect classification and crop damage estimation, weed classification and pressure estimation, fruit identification and yield estimation, decision on replanting (Whigham et al., 2000), and disease identification and disease damage estimation in real time using actual field images collected by AICropCAM.

AICropCAM shows excellent potential in enhancing crop management through crop monitoring. However, the current demonstration requires significant improvements on both hardware and software fronts. Customized circuitry and modular design are required to put AICropCAM in commercial farm applications. The full potential of the AICropCAM can be achieved by putting this camera on a moving platform like a center pivot with a GPS receiver to generate spatiotemporal data. Crop classification must include more crop types, and segmentation models need training data from other crop types. The DCNN models for weed and insect identification require the capability to identify

different weed types, their growth stage, different insect types, and their growth stages to generate effective pest control decisions.

Additionally, improving the models' accuracy in image classification, segmentation, and object detection is crucial. It can be achieved by increasing the number of training image data sets. We also planned to expand the research for multiple edge architecture evaluation. Architectures such as a high-performance edge computer that accepts images from multiple edge devices through short-range, high-speed communication (e.g., Wi-Fi) and can run more accurate deep learning models with higher numbers of parameters, might be a better solution for the primary objectives addressed in this paper. We aim to expand the AICropCAM applications to other crops beyond corn and soybean. By making these improvements, AICropCAM will become a more effective tool for crop management, potentially revolutionizing how we grow and manage crops.

Funding

This work was supported by the United States Department of Agriculture – National Institute of Food and Agriculture grants [Award 2020-68013-32371 to YG and GB, Award 2021-67021-34417 to YG].

CRediT authorship contribution statement

Nipuna Chamara: Methodology, Software, Visualization. **Geng Bai:** Conceptualization, Methodology, Resources. **Yufeng Ge:** Conceptualization, Resources, Supervision, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Nipuna Chamara, Yufeng Ge, Geng Bai has patent pending to University of Nebraska-Lincoln.

Data availability

Data will be made available on request.

Acknowledgements

Jianxin Sun assisted in developing the imaging device with Raspberry Pi Zero used for image acquisition. David Scoby helped the field management and AICropCAM installation. Junxiao Zhang supported the field installation of AICropCAM and smart-phone based acquisition of crop images with insects.

References

Aasen, H., Kirchgessner, N., Walter, A., Liebisch, F., 2020. PhenoCams for field phenotyping: using very high temporal resolution digital repeated photography to investigate interactions of growth, phenology, and harvest traits. Front. Plant Sci. 11 (June), 1–16. https://doi.org/10.3389/fpls.2020.00593.

Anubha, P.S., Sathiesh Kumar, V., Harini, S., 2019. A study on plant recognition using conventional image processing and deep learning approaches. J. Intell. Fuzzy Syst. 36 (3), 1997–2004. https://doi.org/10.3233/JIFS-169911.

ArduCAM ESP8266 UNO board User Guide (pp. 0–9). (2016). www.ArduCAM.com.

Bai, G., Ge, Y., Scoby, D., Leavitt, B., Stoerger, V., Kirchgessner, N., Irmak, S., Graef, G., Schnable, J., Awada, T., 2019. NU-Spidercam: A large-scale, cable-driven, integrated sensing and robotic system for advanced phenotyping, remote sensing, and agronomic research. Comput. Electron. Agric. 160 (March), 71–81. https://doi.org/10.1016/j.compag.2019.03.009.

Barbedo, J.G.A., 2014. Using digital image processing for counting whiteflies on soybean leaves. J. Asia Pac. Entomol. 17 (4), 685–694. https://doi.org/10.1016/j.aspen.2014.06.014.

Cao, K., Liu, Y., Meng, G., Sun, Q., 2020. An Overview on Edge Computing Research. IEEE Access 8, 85714–85728. https://doi.org/10.1109/ACCESS.2020.2991734.

Chamara, N., Alkhadi, K., Jin, H., Bai, F., Samal, A., Ge, Y., 2021. A deep convolutional neural network based image processing framework for monitoring the growth of soybean crops. 2021 ASABE. Annual International Meeting 2100259. https://doi.org/10.13031/aim.202100259.

Chamara, N., Islam, M.D., Bai, G.F., Shi, Y., Ge, Y., 2022. Ag-IoT for crop and environment monitoring: Past, present, and future. Agr. Syst. 203, 103497 https://doi.org/10.1016/j.agsy.2022.103497.

Chamara, N., 2021. Development of an Internet of Things (IoT) Enabled Novel Wireless Multi-Sensor Network for Infield Crop Monitoring. Master's Thesis. Department of Biological Systems Engineering, University of Nebraska-Lincoln.

Datasheet Raspberry Pi Model B, 2019. https://datasheets.raspberrypi.org. Accessed 11 November 2023.

Firdaus-Nawi, M., Noraini, O., Sabri, M.Y., Siti-Zahrah, A., Zamri-Saad, M., Latifah, H., 2018. DeepLabv3+_Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 801–818.

Ghorai, A.K., Barman, A.R., Chandra, B., Viswavidyalaya, K., Jash, S., Chandra, B., Viswavidyalaya, K., Chandra, B., Viswavidyalaya, K., 2021. Image processing based detection of diseases and nutrient deficiencies in plants. SATSA Mukhapatra 25 (1), 1–24.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition kaiming. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778. doi: 10.1002/chin.200650130.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86(11), 2278–2323. doi: 10.1109/5.726791.

Liang, W. Z., Oboamah, J., Qiao, X., Ge, Y., Harveson, B., Rudnick, D. R., Wang, J., Yang, H., Gradiz, A., 2023. CanopyCAM – an edge-computing sensing unit for continuous measurement of canopy cover percentage of dry edible beans. Comput. Electron. Agric. 204 (January), 107498. https://doi.org/10.1016/j.compag.2022.107498.

Luis, S., Filipe, N.S., Paulo, M.O., Pranjali, S., 2020. Deep Learning applications in agriculture: a short review. Deep Learning Applications in Agriculture: A Short Review, 1092 AISC(January), C1. doi: 10.1007/978-3-030-35990-4.

Meidas Trail Cameras, 2022. https://www.meidase.com/product-category/trail-cameras/. Accessed 11 November 2023.

Mistry, S., 2016. Arduino LoRa. MIT License. https://github.com/sandeepmistry/arduino-LoRa. Accessed 11 November 2023.

Park, Y., Krell, R.K., Carroll, M., 2007. Theory, technology, and practice of site-specific insect pest management. J. Asia Pac. Entomol. 10 (2), 89–101.

Paymode, A.S., Malode, V.B., 2022. Transfer learning for multi-crop leaf disease image classification using convolutional neural network VGG. Artif. Intell. Agric. 6, 23–33. https://doi.org/10.1016/j.aiia.2021.12.002.

Richardson, A.D., 2019. Tracking seasonal rhythms of plants in diverse ecosystems with digital camera imagery. New Phytol. 222 (4), 1742–1750. https://doi.org/10.1111/nph.15591.

Sakamoto, T., Gitelson, A.A., Nguy-Robertson, A.L., Arkebauer, T.J., Wardlow, B.D., Suyker, A.E., Verma, S.B., Shibayama, M., 2012. An alternative method using digital cameras for continuous monitoring of crop status. Agric. For. Meteorol. 154–155, 113. https://doi.org/10.1016/j.agrformet.2011.10.014.

Sharma, P., Berwal, Y.P.S., Ghai, W., 2020. Performance analysis of deep learning CNN models for disease detection in plants using image segmentation. Inf. Process. Agric. 7 (4), 566–574. https://doi.org/10.1016/j.inpa.2019.11.001.

Sritarapipat, T., Rakwatin, P., Kasetkasem, T., 2014. Automatic rice crop height measurement using a field server and digital image processing. Sensors (Switzerland) 14 (1), 900–926. https://doi.org/10.3390/s140100900.

Taylor, S.D., Browning, D.M., 2022. Classification of daily crop phenology in phenocams using deep learning and hidden markov models. Remote Sens. (Basel) 14 (2), 1–22. https://doi.org/10.3390/rs14020286.

Tetila, E.C., Machado, B.B., Astolfi, G., Belete, N.A.S., Amorim, W.P., Roel, A.R., Pistori, H., 2020. Detection and classification of soybean pests using deep learning with UAV images. Computers and Electronics in Agriculture, 179(May). doi: 10.1016/j.compag.2020.105836.

Tetila, E.C., MacHado, B.B., Menezes, G.V., De Souza Belete, N.A., Astolfi, G., Pistori, H., 2020b. A deep-learning approach for automatic counting of soybean insect pests. IEEE Geosci. Remote Sens. Lett. 17 (10), 1837–1841. https://doi.org/10.1109/LGRS.2019.2954735.

The MathWorks, I., 2022. MATLAB Coder - MATLAB. MathWorks. https://www.mathworks.com/products/matlab-coder.html.

Tian, H., Wang, T., Liu, Y., Qiao, X., Li, Y., 2020. Computer vision technology in agricultural automation—a review. Inf. Process. Agric. 7 (1), 1–19. https://doi.org/10.1016/j.inpa.2019.09.006.

van Dijk, M., Morley, T., Rau, M.L., Saghai, Y., 2021. A meta-analysis of projected global food demand and population at risk of hunger for the period 2010–2050. Nat. Food 2 (7), 494–501. https://doi.org/10.1038/s43016-021-00322-9.

Wang, Q., Cheng, M., Huang, S., Cai, Z., Zhang, J., Yuan, H., 2022b. A deep learning approach incorporating YOLO v5 and attention mechanisms for field real-time detection of the invasive weed Solanum rostratum Dunal seedlings. Comput. Electron. Agric. 199 (July), 107194 https://doi.org/10.1016/j.compag.2022.107194.

Wang, J., Gao, Z., Zhang, Y., Zhou, J., Wu, J., Li, P., 2022a. Real-time detection and location of potted flowers based on a ZED camera and a YOLO V4-tiny deep learning algorithm. Horticulturae 8 (1). https://doi.org/10.3390/horticulturae8010021.

Wang, Y., Wang, D., Shi, P., Omasa, K., 2014. Estimating rice chlorophyll content and leaf nitrogen concentration with a digital still color camera under natural light. Plant Methods 10 (3), 273–286. https://doi.org/10.1016/S0378-4290(99)00063-5.

Wang, A., Zhang, W., Wei, X., 2019. A review on weed detection using ground-based machine vision and image processing techniques. Comput. Electron. Agric. 158 (January), 226–240. https://doi.org/10.1016/j.compag.2019.02.005.

Whigham, K., Farnham, D., Lundvall, J., Tranel, D., 2000. Soybean replant decision. Department of Agronomy, Iowa State University.

Yasrab, R., Zhang, J., Smyth, P., Pound, M.P., 2021. Predicting plant growth from time-series data using deep learning. Remote Sens. (Basel) 13 (3), 1–17. https://doi.org/10.3390/rs13030331.

Yuan, W., Wijewardane, N.K., Jenkins, S., Bai, G., Ge, Y., Graef, G.L., 2019. Early prediction of soybean traits through color and texture features of canopy RGB imagery. Sci. Rep. 9, 14089. https://doi.org/10.1038/s41598-019-50480-x.

Zualkernan, I., Dhou, S., Judas, J., Sajun, A.R., Gomez, B.R., Hussain, L.A., 2022. An IoT system using deep learning to classify camera trap images on the edge. Computers 11 (1), 1–24. https://doi.org/10.3390/computers11010013.