Pest detection for Precision Agriculture based on IoT Machine Learning application

Andrea Albanese, Donato d'Acunto, and Davide Brunelli

Department of Industrial Engineering,
University of Trento,
Via Sommarive 9, 38123 Povo TN, Italy
{name.surname}@unitn.it

Abstract. Apple orchards are widely expanding in many countries of the world, and one of the major threats of these fruit crops is the attack of dangerous parasites such as the Codling Moth. IoT devices capable of executing machine learning applications in-situ offer nowadays the possibility of featuring immediate data analysis and anomaly detection in the orchard. In this paper, we present an embedded electronic application that automatically detects the Codling Moths from pictures takes by a camera on top of the insects-trap. Image pre-processing, cropping, and classification are done on a low-power platform that can be easily powered by a solar panel energy harvester. The proposed system is assessed in terms of the accuracy of pest recognition and analysis of power consumption for achieving the energy-neutral balance.

Keywords: Internet of Things, Machine Learning, Precision Agriculture

1 Introduction

Electronics and ICT technologies are gaining momentum in agriculture services. Precision farming is developing new solutions for pest detection, water management, treatments optimization nowadays; since the goal of precision agriculture is to get the most healthy product sustainably. Most of these applications use the effort of smart sensors which are managed from low cost and low power embedded systems. Usually, after sensing the surrounding environment, the system does not take any decision about the acquired data and it is transmitted to remote servers for supports. The main drawback of this approach is a large amount of data to be transmitted that hampers scalability of such a distributed paradigm. The key idea is to shift processing near the sensors and finally transmit a report of a few bytes. Moreover, machine learning can improve the performance of a precision agriculture application. This type of algorithms can detect and classify parasites, diseases, and weeds in a very fast way.

This paper is focused on a smart application that detects automatically dangerous parasites for apple orchards, the *Codling Moth*. This insect looks like a butterfly and it is the main problem for apple orchards. Thanks to an insect glue

trap it is possible to take a picture and classify if there are any *Codling Moth* and finally send a notification to the farmer. The classification is done near sensor thanks to a specific low cost and low power hardware, and an energy-efficient solution is proposed to sustain the system as long as possible.

1.1 IoT system

The system is composed of a trap that looks like a little house as shown in the Fig. 1, where a pheromone bait and a glue layer capture the attracted insects even at low-density presence. The farmer usually takes periodic inspections of the traps or mount a wireless camera that sends wirelessly the captured pictures for remote evaluation. This process is expensive and time consuming for the farmer. The proposed work detects the presence of the parasites thank to a machine learning approach sends only notifications of threats to the farmer with the position of them.

(a) Commercial trap.

(b) Prototype of the IoT neural network Codling Moth smart trap.

Fig. 1: Codling Moth smart trap.

The workflow of the proposed application is summarized in Fig. 2. A camera takes pictures inside the trap periodically, the board detects and crops new insects not yet analyzed for the classification. Eventually, a notification is sent to the farmer about the detection of parasites.

For this purpose, the hardware showed in Fig. 3 is based on a Raspberry Pi3 with a Pi Camera. It is in charge of image pre-processing and cropping, whereas a Movidius Neural Compute Stick (NCS) that provides the Intel Myriad X neural accelerator, performs the classification stage. Classification is done by a machine learning algorithm that uses a CNN model tailored for the NCS. The uncommon feature of this IoT application is that the classification stage is done in-situ (near the camera). The processing results, consisting of few bytes after the classification, are transmitted using long-range and a low power communication like LoRaWAN. Thanks to its technical specifications the end nodes can send

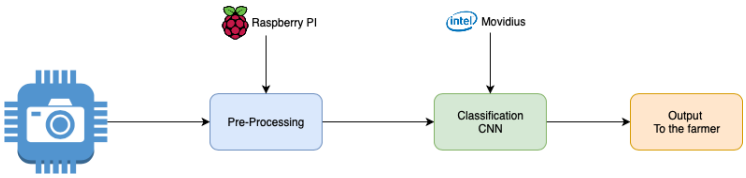


Fig. 2: Flowchart of the system application.

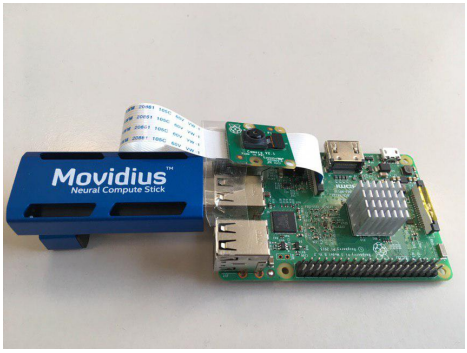


Fig. 3: Hardware implementation.

data in a range of 15 km; additionally, LoRaWAN guarantees the integrity of the transmitted data thanks to its protocol that is covered by security encryption.

1.2 Image pre-processing and deep learning

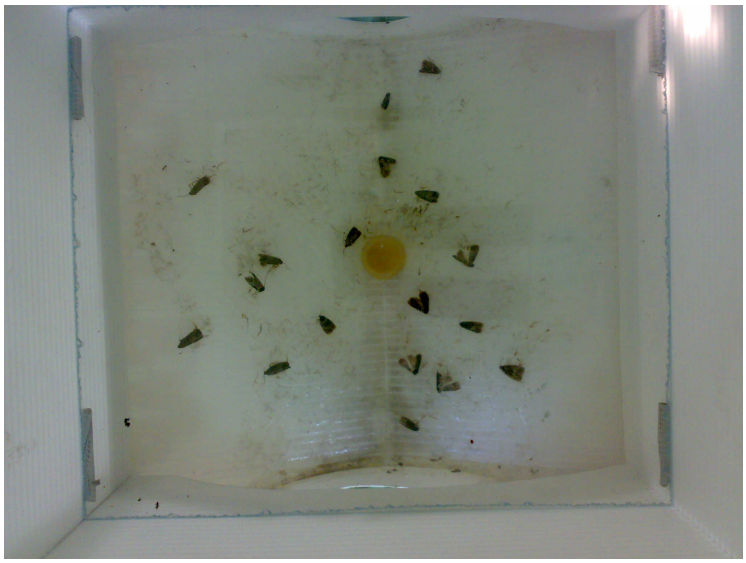
Deep learning is a class of algorithms widely used in machine learning. The network implemented in this project is, in particular, a Convolutional Neural Network. This type of networks are widely used in image classification and object recognition problems. Before training stage of the DNN, a clear and quite large dataset of pictures is necessary to build up the network in an optimal way. The Dataset generation stage is fundamental for supervised methods, and each image used for training and validation stages is known and labeled a priori. This implies that a good dataset for the pictures used during training is crucial for final performance. The dataset generation session started with a small set of pictures (approximately 1300) that has been incremented when more insects have been trapped during the experiment. The dataset is divided into two classes: *codling moth* and *general insects*. For this specific task a VGG16 model, developed by the Oxford University, is used [7]. Then the model is converted to a graph model used to perform the classification on the VPU.

The camera captures the floor of the insect trap, as shown in Figure 4, pictures may contain a high number of insects to classify. Thus, the images are processed to extract each insect in sub tiles from the original taken picture. The task is developed in order to extract easily features like color (a dark subject

on white background) and the shape of the insects through a Blob Extraction algorithm. The process for image crop consists in:

- Conversion of the frame from RGB to GRAY scale;
- Smoothing (or blurring) of the frame with a Gaussian filter;
- Edge extraction through Canny operator;
- Some dilation and erosion of the picture.

After the application of this morphological operators, the blobs are detected through the OpenCV blob detector. The blobs extracted are collected individually in a vector as a rectangle and, from the original frame, each of the corresponding region of interest (RoI) is cropped. All the new pictures are finally saved for the neural network. The all procedure is repeated only for the cropped images that contain more than one blob, in order to enlarge the dataset.



(a) Row picture.

(b) Cropped Codling Moth.

(c) Cropped general insect.

(d) Cropped Codling Moth.

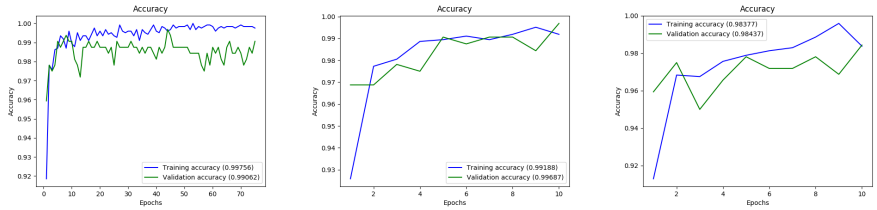
Fig. 4: Examples of pre processed images.

2 Training, Validation and Test

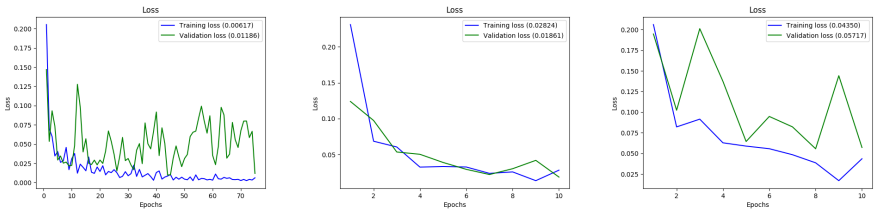
For the training stage, we use the effort of the rapid development of neural networks for image classification based on TensorFlow library [4].
The training stage is an offline process aiming to optimize the neural network through a large dataset of labeled images as said before. Therefore the system can learn which category the images belong to. The basic element of a DNN is the neuron (or node). It is multiplied by a so called weight value only when the input is ready. For example, if a neuron has four inputs, it has four weight values which can be adjusted during the training time. A DNN could be improved through many parameters involved in the process, in our case the most important parameters that change the performance in a significant way are the number of epochs and image size. The first is how many times the entire set of training vectors is used to update the weights, at the end of each epoch a validation step is computed to evaluate the ongoing training process. So the objective is to find a good tradeoff between the two parameters described before in order to complete as good as possible the training stage and even to meet the hardware constraints. In our application the following three different configurations was used:

- 75 epochs, image size 224×224;
- 10 epochs, image size 112×112;
- 10 epochs, image size 52×52;

In Fig. 5 is possible to look at the results obtained in the training tests.



(a) 75 epochs, image size 224x224. (b) 10 epochs, image size 112x112. (c) 10 epochs, image size 52x52.



(d) 75 epochs, image size 224x224. (e) 10 epochs, image size 112x112. (f) 10 epochs, image size 52x52.

Fig. 5: Training and validation accuracy and loss function.

Notice that training and validation accuracy using 75 epochs is going to be saturated. This means that the network does not provide enough accuracy during the test stage and is not able to generalize as good as required.

Thus the epochs can be decreased to achieve better results, as shown in the graphs, 10 epochs are enough for good accuracy. Moreover, in order to avoid possible overflow and to save memory on the Raspberry Pi 3, the image size is decreased to work with a simpler model and to meet the hardware constraints. Image size of 112x112 and 52x52 have been tested and used. The chosen image size shows worse performance with respect to the ones obtained using bigger image size, the accuracy achieved is, nevertheless, 98% which satisfy quietly the requirements for an IoT system for parasites monitoring. After the training and validation stage is finished, the next step is to compute a test through a new set of data (a subset of the original dataset), which is never seen by the DNN, this in order to assess the performance and the generalization of the network. This step is crucial to confirm the accuracy computed during validation.

In Fig. 6 is possible to look at an example of the output from the classification stage. Our DNN provides a measure of the confidence which indicates how the detected insect is close to a general insect or the *Codling Moth*.
The tests were done in an apple orchard during 12 weeks, with the insect glue trap shown in Fig. 1. Classification results are summarized as follows:

- 80,6% was classified correctly;
- 4,8% was false positives;
- 6,4% was false negatives;
- 8,1% was uncertain;

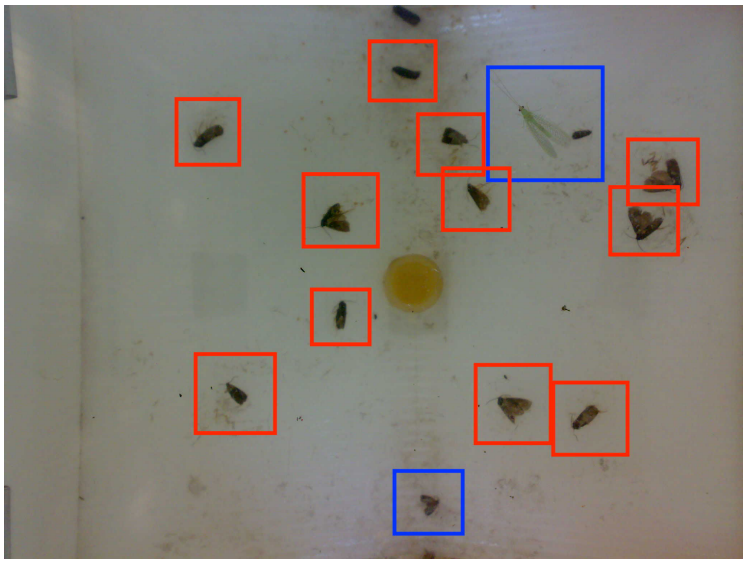


Fig. 6: Example of moth detection from the system.

This IoT application will be executed twice per day. The overall application workflow is divided into five general tasks consisting of:

- Task 0: boot of Raspberry;
- Task 1: camera capture;
- Task 2: preprocessing;
- Task 3: classification;
- Task 4: report/alarm generation;

Moreover it is measured the current consumption and the duration of each task and they are summarized in the following table:

Task	**Period (s)**	**Average Current (mA)**
T0	43,68	345
T1	3,45	394
T2	4,07	501
T3	10,19	525
T4	0,34	525

Table 1: Overview table of the consumption of each task.

A real time clock (RTC) is used to power the IoT application on when planned, and, when the taskset is completed, the system shuts down. In Fig. 7 the overall power consumption from T0 to T4 is shown. It is possible to observe that Task T3 is the most power-hungry task, because it uses the Raspberry with the Intel Movidius NCS neural accelerator. The total energy necessary to sustain an application cycle is 124.1 J.

3 Conclusions

The average power consumption of the proposed system is very small, because of its low duty cycle. Since the system is activated twice per day for a few minutes, the average current consumption is only of few μA. In fact, the IoT smart trap consumes only 248.2 J/day; and a 9000 mAh battery is sufficient to sustain the system for more than 1 year. An additional 0,5W solar panel of few hundreds cm^2, would furtherly extend the energy autonomy, until the full sustainability where energy intake is enough to operate unattended indefinitely. This particular feature represents a breakthrough for agricultural activities, because this means that a farmer could use smart IoT insect trap forgetting about its maintenance, and waiting for only automatic alerts if some *Codling Moth* is captured. Due to the low cost of the components, the proposed work can scale to several installations in the farmer's apple orchard, and save time and money for human intervention in trap checking every day. This type of application is innovative because it is possible to use treatments for *Codling Moth* only when really needed,

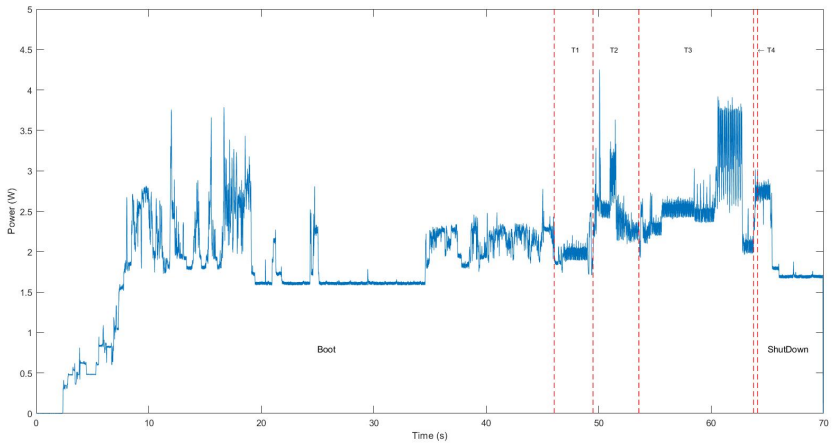


Fig. 7: System power consumption.

saving over usage of chemicals treatments, and mitigating their impact on the environment.

References

1. Machine Learning in Agriculture: Applications and Techniques, `https://medium.com/sciforce/machine-learning-in-agriculture-applications-and-techniq\-ues-6ab501f4d1b5`. Last accessed 15 May 2019
2. Intel Movidius Neural Compute SDK Documentation, `https://movidius.github.io/ncsdk/index.html`. Last accessed 15 May 2019
3. Raspberry Pi Documentation, `https://www.raspberrypi.org/documentation/`. Last accessed 15 May 2019
4. frank1789/NeuralNetworks, `https://github.com/frank1789/NeuralNetworks`. Last accessed 25 May 2019
5. Adelantado, F., Vilajosana, X., Tuset-Peiro, P., Martinez, B., Melia-Segui, J., Watteyne, T.: Understanding the limits of lorawan. IEEE Communications Magazine **55**(9), 34–40 (Sep 2017).
6. Ding, W., Taylor, G.: Automatic moth detection from trap images for pest management. Comput. Electron. Agric. **123**(C), 17–28 (Apr 2016).
7. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)