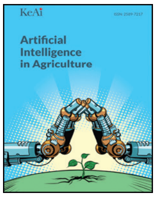
Artificial Intelligence in Agriculture 6 (2022) 129–137

Contents lists available at ScienceDirect

Artificial Intelligence in Agriculture

journal homepage: http://www.keaipublishing.com/en/journals/artificial-intelligence-in-agriculture/



Developing a multi-label tinyML machine learning model for an active and optimized greenhouse microclimate control from multivariate sensed data

Ilham Ihoume *, Rachid Tadili, Nora Arbaoui, Mohamed Benchrifa, Ahmed Idrissi, Mohamed Daoudi

Solar Energy and Environment Laboratory, Mohammed V University in Rabat, Faculty of Sciences, B.P 1014, Rabat, Morocco

ARTICLE INFO

Article history:
Received 24 May 2022
Received in revised form 18 August 2022
Accepted 30 August 2022
Available online 5 September 2022

Keywords:
Agricultural greenhouse
Microclimate control
Machine learning
Optimization
TinyML

ABSTRACT

In the uncertainties within which the worldwide food security lies nowadays, the agricultural industry is raising a crucial need for being equipped with the state-of-the-art technologies for a more efficient, climate-resilient and sustainable production. The traditional production methods have to be revisited, and opportunities should be given for the innovative solutions henceforth brought by big data analytics, cloud computing and internet of things (IoT). In this context, we develop an optimized tinyML-oriented model for an active machine learning-based greenhouse microclimate management to be integrated in an on-field microcontroller. We design an experimental strawberry greenhouse from which we collect multivariate climate data through installed sensors. The obtained values' combinations are labeled according to a five-action multi-label control strategy, then used to prepare a machine learning-ready dataset. The dataset is used to train and five-fold cross-validate 90 Multi-Layer Perceptrons (MLPs) with varied hyperparameters to select the most performant –yet optimized– model instance for the addressed task. Our multi-label control approach enables designing highly scalable models with reduced computational complexity, comprising only n control neurons instead of $(1+\sum_{n}^{k=1} C_n^k)$ neurons (usually generated from a classic single-label approach from n input variables). Our final selected model incorporates 2 hidden layers with 7 and 8 neurons respectively and 151 parameters; it scored a mean accuracy of 97% during the cross-validation phase, then 96% on our supplementary test set. The model enables an intelligent and autonomous greenhouse management with the less required computations. It can be efficiently deployed in microcontrollers within real world operating conditions.

1. Introduction

In its most recent "Food Security and Nutrition in the World" annual report, the United Nations Food and Agriculture Organization (FAO) highlighted substantial ongoing deviations with respect to the commitments made towards ending world hunger and malnutrition by 2030. These deviations have been amplified even more by the pandemic advent and its persistent effects, which still impact severely the worldwide food security (Lopez-Ridaura et al., 2021). The latter –in conjunction with the actual supply chain disturbances and the recent worldwide geopolitical movements– is threatened by another major risk, which is climate change and its induced global warming effect (Lal, 2021). Crops are tightly contingent on climatic factors such as the ambient temperature, the relative humidity, the received solar radiation, the wind as well as the different forms of precipitations such as rain, snow and hail (Iizumi and Ramankutty, 2015; Liao et al., 2021). As a consequence, agriculture is one of the most affected sectors by climate change, whose impact grows increasingly and compromises various harvest types (Doshi and Varghese, 2022). In this context, the urgent arising necessity is seeking for new innovative techniques for more climate-resilient –yet productive– agricultural models, and one of the most promising solutions towards this issue is the use of agricultural greenhouses (Cordovil et al., 2020; Azadi et al., 2021). The prominent advantage of these greenhouses is making it possible for farmers to regulate the environmental growing variables actively and producing –regardless of the outdoor climatic conditions– various crops at any year season. In this way, crops grow in optimal conditions and remain protected from external factors that may harm them (Ihoume et al., 2022; Katzin et al., 2022). Nowadays, greenhouse design and development are witnessing significant advances. The state-of-the-art "smart" models are equipped with sensors that collect consistently

* Corresponding author.
E-mail address: ilham.ihoume@um5r.ac.ma (I. Ihoume).

https://doi.org/10.1016/j.aiia.2022.08.003

environmental data, along with actuators that can perform indoor actions accordingly such as heating, lighting, ventilation, CO_2 enrichment and irrigation (Angelopoulos et al., 2020; Jamil et al., 2022).

Smart greenhouses are promising; however, they generate important climatic data streams that are not easily manageable (Escamilla-García et al., 2020). Managers face difficulties in analyzing, understanding and exploiting these continuous streams correctly. As a consequence, they may misunderstand how climatic factors correlate with their crops, and thus to deploy timely and efficient corresponding management strategies to maximize the production (Azaza et al., 2016). Conventionally, climatic data is collected from indoor greenhouse sensors then sent to the farmers who have to set the corresponding control actions continuously such as lighting, ventilation and irrigation. In this process, competent and experienced managers are needed, which is not always achievable (Mani and Sankaranarayanan, 2017). Moreover, this classic decision-making scheme is not optimized, since it does not take full advantage of the interdependencies between the collected variables and the underlying correlations with the crops' states of growth.

2. Related work

Many research works have focused on developing intelligent solutions for greenhouse microclimate management (See Table 1). Research and development efforts focus on proposing performant algorithms, as well as reliable and interoperable hardware implementations in real-world operating conditions. The objective is achieving intelligent decision-making solutions that make it possible for greenhouses to operate autonomously, and thus improving the quality and yield of harvests (Alwis et al., 2022). Revathi and Sivakumaran (2016) developed a fuzzy logic-based temperature controller that regulates the internal greenhouse temperature based on live temperature monitoring from an internal sensor. Azaza et al. (2016) also developed a fuzzy logic-based greenhouse climate controller operating on humidity, temperature, CO_2 and illuminance to perform indoor control-related actions. However, in addition that the first work tackled only one variable i.e., the ambient temperature, the major drawback of fuzzy logic-based systems is being completely dependent on human knowledge and expertise. Angelopoulos et al. (2020) developed a reinforcement learning-based system targeting smart irrigation within a strawberry greenhouse in Greece. They found that the system outperforms conventional irrigation in terms of soil moisture variation and water consumption; however, it manages only one task that is irrigation.

In the hydroponics subfield, Alipio et al. (2017) proposed a Bayesian Network (BN) algorithm to automate plant growth by controlling the light intensity, the soil acidity, the water temperature and the humidity. They were able to increase the production by around 66.67% compared to manual control; however, from a probabilistic perspective BNs generate probability density functions rather than the maximum a posteriori estimated direct parameter values, which adds additional unnecessary complexity to the system. Mehra et al. (2018) proposed a Multi-layer Perceptron (MLP) neural network to monitor and manage the growth of tomatoes. An accuracy of 88% was achieved; however, for only 3 input variables the proposed model involved $(1 + \sum_{3}^{k=1} C_{3}^{k} = 8)$ control neurons at its output layer, which makes it practically unscalable for more complex scenarios.

3. Context, contribution and organization of the work

The aforementioned works underline several efforts made in the context of moving towards smart greenhouse microclimate control through case studies. However, the reviewed solutions/approaches present at least one of the following drawbacks:

- Being mono-task i.e., handling only one action in function of a single climatic variable such as the temperature. While regulating one single variable ensures its adequacy with a particular crop requirement, it remains insufficient since crops usually correlate with multiple variables simultaneously (Poudel and Shaw, 2016; Liu et al., 2020).
- Being not fully autonomous and intelligent, with involving continuous manual intervention to set up the corresponding control actions to be taken inside a greenhouse, depending on the collected climatic data.
- Achieving a performance below the threshold of being concretely deployable to on-field application in real world operating scenarios.
- Being developed with no optimization for an implementation within on-field micro-controllers, though it is a crucial aspect within the addressed thematic. Machine and deep learning were initially deployed in big data centers that are equipped with highly performant computers, which can handle huge amounts of parallel computations. However, there is actually a rising need for implementing machine/deep learning models within physical and portable devices for more practical and interactive applications. Such devices have very limited memory and processing power, thus requiring simpler –yet efficient– models to be designed (Dutta and Brahali, 2021; Frankowiak et al., 2005). The machine learning subfield that addresses this problematic is called Tiny Machine Learning (TinyML) (Ray, 2022; Han and Siebert, 2022).

Greenhouses are complex and non-linear systems (Escamilla-García et al., 2020). An efficient climate-based management for these systems require algorithms that can autonomously learn the underlying patterns within the collected data then map them to the most optimized control actions. Additionally, these algorithms have to be light enough for a further concrete implementation in on-field microcontrollers. In this context, we develop a model for microclimate control of a strawberry agricultural greenhouse with the following contributions:

- In a machine learning-based approach, we use an MLP architecture to achieve full autonomy with no required manual intervention.
- In a tinyML-oriented approach, we conduct a thorough and systematic hyperparameter tuning with extensive performance evaluation in the chosen architecture in order to:
  - o Achieve the highest performance possible for the addressed task.
  - o Obtain a lightweight model in terms of parameters and internal computations for a further implementation in a microcontroller within real world operating conditions.
- We propose a multi-label control strategy that shrinks dramatically the number of required control neurons from $(1 + \sum_{n}^{k=1} C_{n}^{k})$ to just n neurons, which enables more computational efficiency and increased generalizability to more complex control scenarios.

The upcoming sections are structured as follows: Section 4 presents the used materials and adopted methodology; Section 5 exposes and

Table 1
Summary of the reviewed works.

	Approach	Input climatic variables	Performance metric
Revathi and Sivakumaran (2016)	Fuzzy-logic	Temperature	–
Azaza et al. (2016)		Humidity, temperature, CO2 and illuminance	–
Angelopoulos et al. (2020)	Reinforcement learning	Soil moisture, water consumption	–
Alipio et al. (2017)	Bayesian network	Light intensity, soil acidity, water temperature and humidity	66.67% (production increase)
Mehra et al. (2018)	Multi-layer Perceptron	Temperature, illuminance, soil moisture	88% (accuracy)

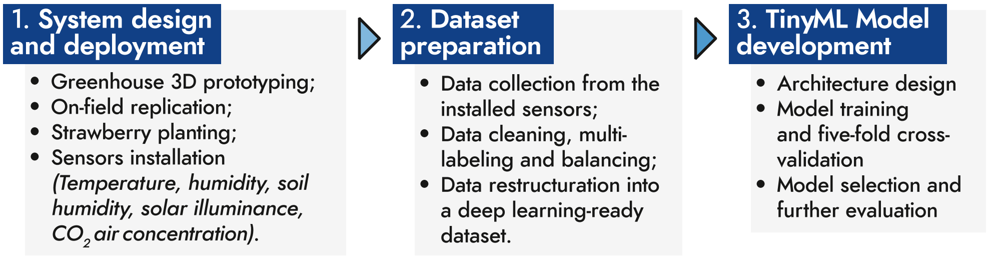

1. System design and deployment
• Greenhouse 3D prototyping;
• On-field replication;
• Strawberry planting;
• Sensors installation (Temperature, humidity, soil humidity, solar illuminance, CO_2 air concentration).
2. Dataset preparation
• Data collection from the installed sensors;
• Data cleaning, multi-labeling and balancing;
• Data restructuration into a deep learning-ready dataset.
3. TinyML Model development
• Architecture design
• Model training and five-fold cross-validation
• Model selection and further evaluation

Fig. 1. Flowchart of the general methodology.

discusses the obtained results; Section 6 concludes the work along with providing recommendations for future research.

4. Materials and methods

The methodology of this work comprises three phases depicted in Fig. 1: (1) the first one consists in the design and deployment of an experimental strawberry greenhouse, equipped with active sensors; (2) the second comprises collecting the sensed data to prepare a machine learning-ready dataset; (3) the third tackles the development, cross-validation of multiple models, then the selection and further test of the optimal one. The details of each phase will be presented in the upcoming subsections.

4.1. System design and deployment

The experimental greenhouse was first designed within a 3D modeling environment (3ds Max), then replicated on the terrace of the Solar Energy and Environment Laboratory at the Faculty of Sciences in Rabat (Morocco) (Altitude: 66 m, Longitude: 6°49′57″, Latitude: 34°00′47″) (Fig. 2). The greenhouse has a base of width 2.30 m, of height 3.20 m and a footprint area of 7.36 m^2. It is elevated from one side by 2.07 m and from the other with 3.20 m, forming a learning roof of width 2.50 m. The greenhouse was equipped with a solar heating system that operates through storing the solar energy collected at daytime and releasing at nighttime to maintain a temperature range around the strawberry temperature set point. To ventilate the ambient air, the greenhouse was equipped with an automatic opening window that uses an electrical motor and a fan with a power of 40 W.

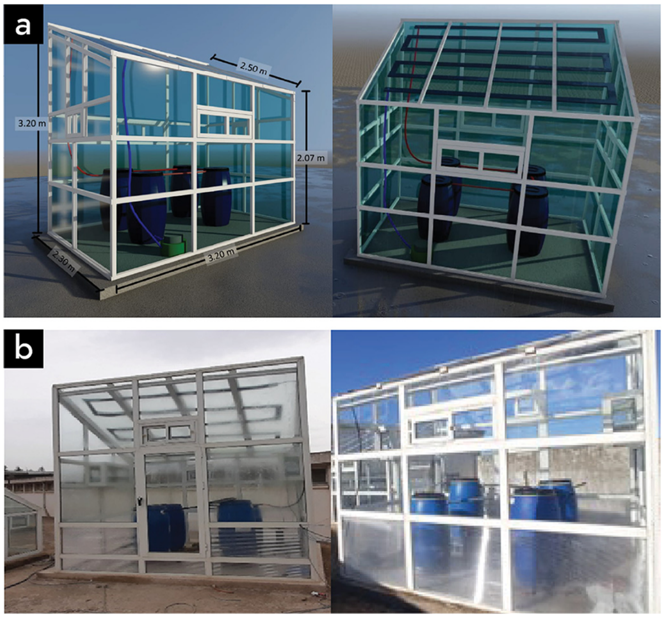


Fig. 2. The experimental greenhouse: (a) A 3D designed prototype using a 3D modeling software (3ds Max); (b) The on-field replicated model.

The greenhouse was installed beside a weather station comprising a pyranometer, a humidity and temperature sensors, a rain gauge, an anemometer and a barometer (Fig. 3). Inside the greenhouse (Fig. 4), a pyranometer was installed to measure the global solar radiation. The relative humidity was measured in the center of the greenhouse at a height of 1 m above the ground using a RH sensor. A T-type thermocouple (copper-constantan) at the interior to measure the interior temperature. All sensors were connected to an acquisition box, where data were recorded at a frequency of 6 s and averaged over a one-hour tile scale before being processed and stored under a tabular format. The technical specifications of each sensor are presented in Table 2.

4.2. Dataset preparation

Five climatic parameters were measured during the winter season of the year 2022 inside the greenhouse: (1) the temperature, (2) the humidity, (3) the soil humidity, (4) the solar illuminance and (5) the air CO_2 concentration. The recorded data were collected from the installed sensors then cleaned from erroneous records. The combinations of the obtained values were normalized to the unit interval with respect to each input variable, then labeled using multi-label vectors with one-hot encoding according to control actions that have to be performed automatically within the greenhouse. Each multi-label is a five-dimensional vector where each binary numerical value is associated with one control action as illustrated in Fig. 5. As examples, a vector such as [0,0,0,0,0] means that no control action has to be taken; a vector such as [1,1,0,0,0] means that the window and the ventilator have to actioned to regulate the indoor temperature and humidity respectively.

The dataset was balanced with respect to each action occurrence. Afterwards, it was split into two sub-datasets: (1) a training/cross-validation sub-dataset of 80%, and (2) a test sub-dataset of 20% held out for further evaluation.

Fig. 3. Weather station of the laboratory of solar energy and environment at the Faculty of Sciences in Rabat (Morocco).

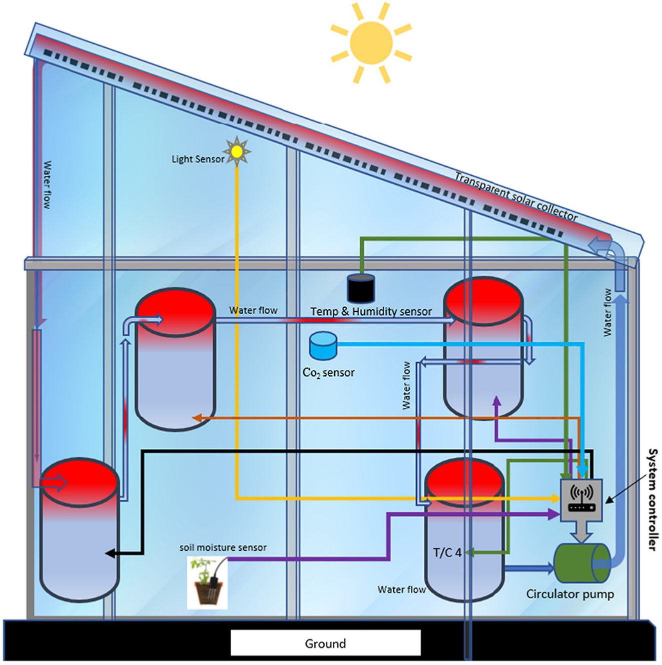


Fig. 4. The sensors' location inside the greenhouse.

Table 2
Technical specifications of the sensors installed in the weather station and inside the greenhouse.

	Sensor	Measuring range	Accuracy	Unit
Weather station	Solar radiation	0–5000	±0,1	W/m^2
	Anemometer	0,4–30	±0,2	m/s
	Outdoor humidity	0–99	±5	%
	Precipitations	–	±0,01	mm
	Pressure	500–1100	±5	hPa
	Outdoor temperature	−40–80	±0,1	°C
Inside the greenhouse	Humidity	0–99	±5	%
	Temperature	−40–80	±0,1	°C
	Soil humidity	0–1000	±0,1	–
	CO$_2$ air concentration	0–5000	±0,4	ppm
	Solar illuminance	0–1000	±3	lux

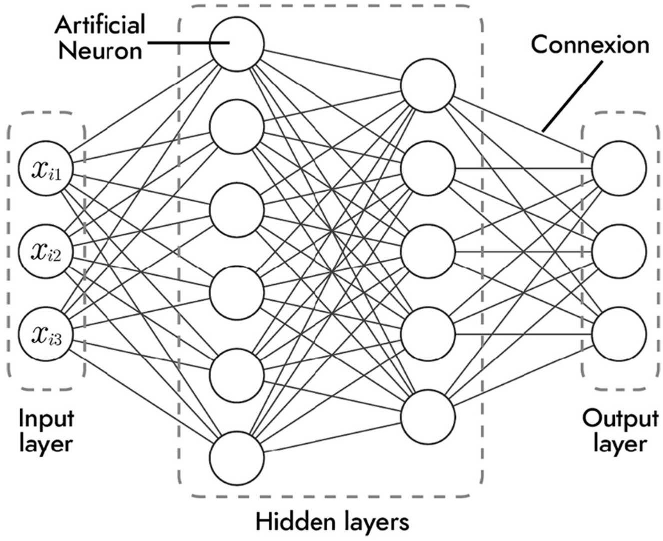


Fig. 6. A multi-layer perceptron with $X \in \mathbb{R}^{(m,3)}$, two hidden layers (with six neurons in the first layer and five in the second) and a three-output prediction layer.

4.3. Model development

4.3.1. Architecture design

In order to develop our autonomous and intelligent greenhouse control solution, we have chosen a machine learning-based Multi-Layer Perceptron (MLP) architecture. MLPs are a class of feed-forward neural networks capable of approximating complex linear/non-linear functions. To achieve that, they rely on linked mathematical predictors called artificial neurons, arranged within three types of stacked layers as depicted in Fig. 6 (Pedrycz and Chen, 2020; Arias del Campo et al., 2021).

- One input layer whose role is transferring to the next layer (called the first hidden layer) a set of m $n-$dimensional input features: $X = \{X_i\}_{i\in[\![1,m]\!]}$ where for each i: $X_i = \{x_{ik}\}_{k\in[\![1,n]\!]}$.
- One hidden layer, which receives the input features then transfers them either to an output layer or an additional hidden one that operates likewise.
- One output layer designed to output the final prediction $\widehat{y}_i$ corresponding to each of the initially received inputs X_i.

An artificial neuron produces from a set of inputs: $X_i = \{x_{ik}\}_{k\in[\![1,n]\!]}$ an activated linear combination a_i expressed as follows: $a_i = f(\sum_{k=1}^{n} w_{ik}x_{ik} + b_i)$ where $\{w_{ik}\}_{k\in[\![1,n]\!]}$ are called the weights of the neuron, b_i its bias and f its activation function (Arias del Campo et al., 2021). As shown in Fig. 6, every neuron is connected to all the neurons in the previous and next layers, this is why the layers of an MLP are also called fully connected layers. Every inter-neuron connection is equivalent to a forward activation transfer; the activations of each layer are therefore considered as the next layer's inputs. The weights and biases of all neurons within an MLP are its parameters; they are first randomly initialized, then optimized (or learned) during a training process (that will be detailed afterwards). The number of layers and neurons per layer as well as the implemented activation functions are called hyperparameters; they are tuned prior to any training session (Alzubaidi et al., 2021).

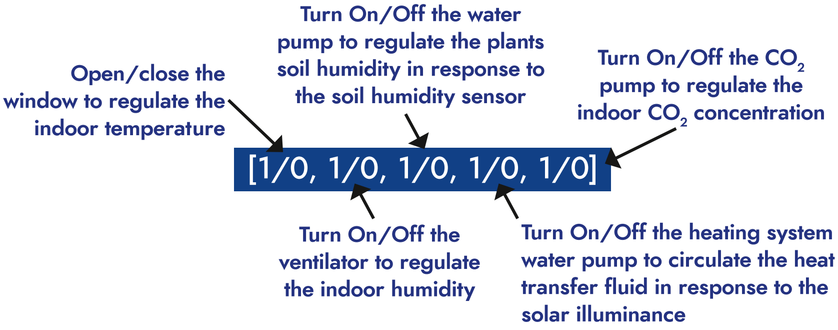

Open/close the window to regulate the indoor temperature
Turn On/Off the water pump to regulate the plants soil humidity in response to the soil humidity sensor
Turn On/Off the CO$_2$ pump to regulate the indoor CO$_2$ concentration
[1/0, 1/0, 1/0, 1/0, 1/0]
Turn On/Off the ventilator to regulate the indoor humidity
Turn On/Off the heating system water pump to circulate the heat transfer fluid in response to the solar illuminance

Fig. 5. The five-action multi-label control strategy.

To find the optimal model architecture for the addressed task, we have started from an initial MLP, comprising one hidden layer of one neuron. Next, we increased progressively the number of layers, along with the number of neurons per layer, to create multiple MLP instances that were trained and cross-validated on the prepared dataset.

4.3.2. Model training and cross-validation

In an MLP, hidden layers learn the underlying patterns of a given dataset through an extraction/combination process of multiple sequentially constructed abstract representations. For this purpose, an MLP has first to be trained on that dataset. The training is an optimization problem, aiming at finding the model's set of parameters that minimize a metric quantifying the difference between the dataset labels $Y = \{y_i\}_{i \in \mathbb{R}^m}$ and the model predictions $\hat{Y} = \{\hat{y}_i\}_{i \in \mathbb{R}^m}$. This metric is called the cost function J (Alzubaidi et al., 2021; Sarker, 2021).

Each obtained model instance was trained on the prepared dataset, in parallel with a five-fold cross validation procedure. The latter consists in training the instance on five-split configurations. In each one, one split (1/5) is reserved to validate the model after being trained on the four other ones, which provides more realistic performance estimates for the evaluation (Refaeilzadeh et al., 2009). A variant of an iterative gradient-based optimization algorithm has been adopted i.e., the Adam optimizer (Kingma and Ba, 2017). For each model instance, the parameters were randomly initialized; afterwards, a series of iterations (or epochs) –whose number is also a hyperparameter– were performed. Each series comprises:

- A forward propagation phase, where –with the last parameter state– the model predictions are computed, and the model is evaluated through calculating the cost function and the model accuracy on the training and validation sub-datasets.
- A backpropagation phase, where each of the model parameters is updated using the formula: $w_i = w_i - \alpha \partial J / \partial w_i$, where α is a hyperparameter called the learning rate.

A Rectified Linear Unit (ReLU) activation function expressed as follows $f : z \rightarrow \max(z, 0)$ was implemented in all the hidden layers' neurons. ReLU is a widely used activation function in machine/deep learning models since it involves less computations compared to other used functions such as sigmoid and tangent hyperbolic (Apicella et al., 2021). Since the model has to output a vector of binary numbers, each output neuron was implemented with a sigmoid activation function that produces activation values comprised between 0 and 1. Training was performed for 250 epochs, using the Adam optimizer and a categorical cross-entropy cost function. Data was transferred to the optimizer through batches of 2048 samples. To track the model performance, the cost function and the accuracy were computed at the end of each epoch on the training and validation sets using the following expressions respectively:

$$J = \sum_{i=1}^{m} \sum_{k=1}^{n} y_k \log(\hat{y}_k)$$

where J is the cost function value, m is the number of elements in the training (or validation) dataset, n the dimension of each element X_i, $\widehat{y}_k$ the k-th scalar value of the model prediction for the dataset element i and y_k the k-th scalar value of its corresponding label.

$$Acc = \frac{tp + tn}{tp + tn + fp + fn}$$

where Acc is the accuracy value, tp and tn are respectively the numbers of true positives and negatives, and likewise, fp and fn are respectively the numbers of false positives and negatives in the training (or validation) datasets.

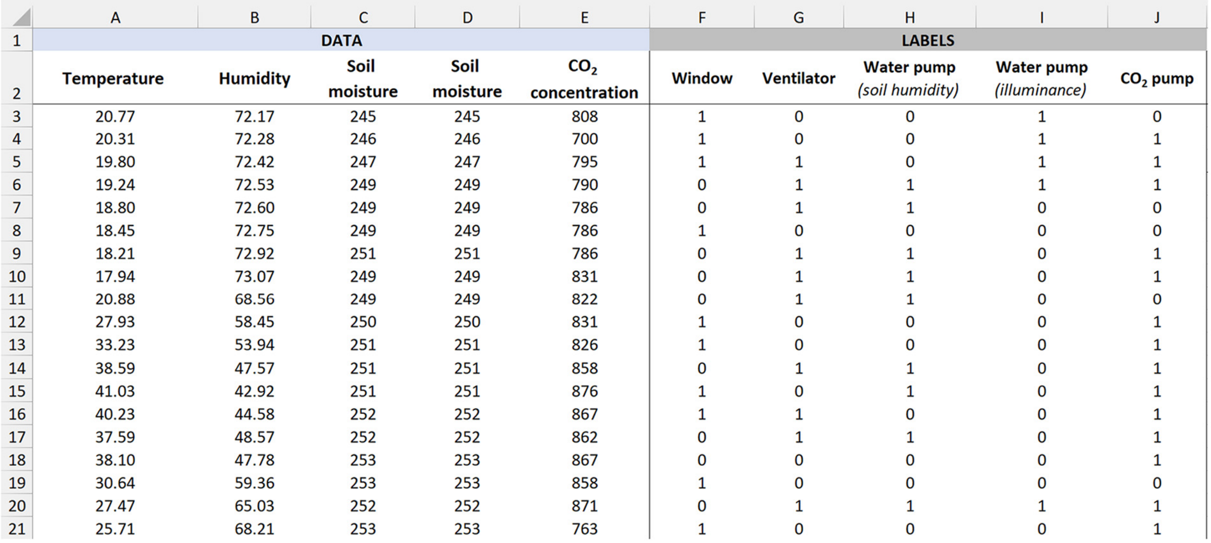
Model prototyping and training were performed within a Python development environment using TensorFlow (version 2.5), an open-source framework for machine/deep learning applications. Computations were carried out using the Google cloud solution Collaboratory, within a machine equipped with the following main components:

- A CPU type Intel Xeon, with a frequency of 2.2 GHz;
- A GPU type Tesla T4, having a memory of 16GB and 585 MHz clock speed;
- 12GB of RAM.

5. Results and discussion

5.1. Data preparation

The data preparation phase yielded a labeled dataset (Fig. 7) of 42,571 records (rows). The final machine learning-dataset comprised 34,056 samples dedicated to train/cross-validate the models, and 8515 samples left for further evaluation of the selected model.



	A	B	C	D	E	F	G	H	I	J
1	DATA					LABELS				
2	Temperature	Humidity	Soil moisture	Soil moisture	CO_2 concentration	Window	Ventilator	Water pump *(soil humidity)*	Water pump *(illuminance)*	CO_2 pump
3	20.77	72.17	245	245	808	1	0	0	1	0
4	20.31	72.28	246	246	700	1	0	0	1	1
5	19.80	72.42	247	247	795	1	1	0	1	1
6	19.24	72.53	249	249	790	0	1	1	1	1
7	18.80	72.60	249	249	786	0	1	1	0	0
8	18.45	72.75	249	249	786	1	0	0	0	0
9	18.21	72.92	251	251	786	0	1	1	0	1
10	17.94	73.07	249	249	831	0	1	1	0	1
11	20.88	68.56	249	249	822	0	1	1	0	0
12	27.93	58.45	250	250	831	1	0	0	0	1
13	33.23	53.94	251	251	826	1	0	0	0	1
14	38.59	47.57	251	251	858	0	1	1	0	1
15	41.03	42.92	251	251	876	1	0	1	0	1
16	40.23	44.58	252	252	867	1	1	0	0	1
17	37.59	48.57	252	252	862	0	1	1	0	1
18	38.10	47.78	253	253	867	0	0	0	0	1
19	30.64	59.36	253	253	858	1	0	0	0	0
20	27.47	65.03	252	252	871	0	1	1	1	1
21	25.71	68.21	253	253	763	1	0	0	0	1

Fig. 7. Snapshot from the prepared dataset.

Table 3
Summary of the constructed model instances of the first group (of one hidden layer) with the total trainable parameter number of each instance.

Number of neurons per hidden layer 1								
1	2	3	4	5	6	7	8	9
16	27	38	49	60	71	82	93	104

5.2. Model development

5.2.1. Architecture design

The core defining hyperparameters within the architecture of an MLP are the number of layers and neurons within each layer, especially the hidden ones. In this study, two groups of models have been constructed: (1) The first with one hidden layer where the number of neurons has been tuned between 1 and 9; (2) The second with two hidden layers where the number of neurons within each layer has also been tuned likewise. The total number of trainable parameters of each group are presented respectively in Tables 3 and 4.

The architecture design phase aimed at finding the model that will score the highest mean accuracy (averaged over the five-fold cross validation procedure) along with being the less complex i.e., with the minimum number of parameters for a further implementation in a microcontroller. Tables 3 and 4 show how increasing the layers' number and the contained neurons generate more parameters. For the case of two-layer MLPs, the modifications induced in the hidden layers operate symmetrically. A tuple of parameters (a,b), chosen respectively for layers 1 and 2, generates identical parameters as the inversed tuple (b,a).

5.2.2. Multi-labeling

As shown in Fig. 8, compared to a traditional labeling approach, multi-labeling enables a huge gain in terms of neurons within the output layer, which in turn, reduces the number of generated parameters by the layer substantially. For 5 climatic input variables (as it is the case in our work), adopting a traditional labeling approach would have resulted in MLPs with 31 neurons in the output layer. This number would have escalated quickly to 1024 neurons for 10 input variables. Based on this, we content that multi-labeling is a highly efficient approach for designing lightweight and scalable models within the addressed thematic.

5.2.3. Model training and validation

For the first and second model groups, the heatmaps in Figs. 9 and 10 illustrate respectively the final mean values of the cost, the validation cost, the accuracy and the validation accuracy during the training/five-fold cross-validation procedure. For Fig. 9, the upper axis represents the number of neurons within the hidden layer. For Fig. 10, it represents the number within hidden layer 2, while the left axis contains the neurons' number per hidden layer 1.

Table 4
Summary of the constructed model instances of the second group (with two hidden layers) with the total trainable parameter number of each instance.

		Number of neurons per hidden layer 2								
		1	2	3	4	5	6	7	8	9
Number of neurons per hidden layer 1	1	18	25	32	39	46	53	60	67	74
	2	25	33	41	49	57	65	73	81	89
	3	32	41	50	59	68	77	86	95	104
	4	39	49	59	69	79	89	99	109	119
	5	46	57	68	79	90	101	112	123	134
	6	53	65	77	89	101	113	125	137	149
	7	60	73	86	99	112	125	138	151	164
	8	67	81	95	109	123	137	151	165	179
	9	74	89	104	119	134	149	164	179	194

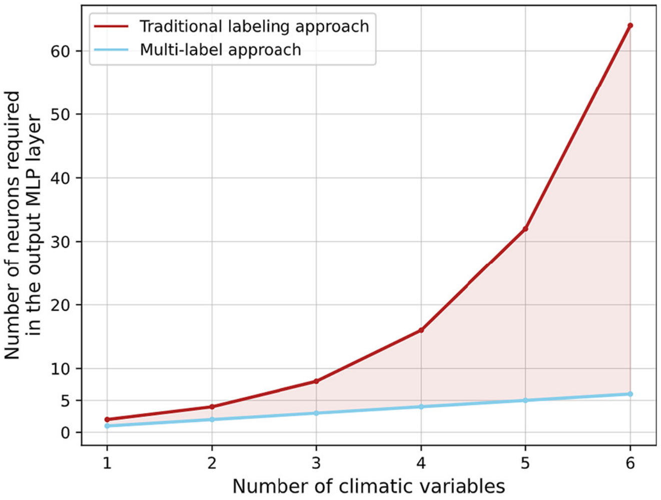


Fig. 8. Multi-labeling versus traditional labeling approach.

For both groups, the models' performance improves with the construction of more complex MLPs. More neurons yield generally better performance and vice versa. This observation is quite logical since having a more complex architecture enables to fit more complex data patterns. On the other hand, trying to fit a very simple model to a complex data distribution yields poor performance, since the trainable parameter number is insufficient. We will denote the model instances within group 1 by *model* (i) where *i* is the number of neurons within the hidden layer. For group 2, each instance will be denoted by *model* (i,j) where *i* and *j* are respectively the number of neurons per hidden layers 1 and 2.

- For the first group of models, the highest validation accuracy value (0.94) was scored by models (8) and (9).
- For the second group of models, the highest validation accuracy value (0.97) was scored by models (7,8) and (8,8).

5.2.4. Model selection and further evaluation

Among the previously cited models, model (7,8) is the most performant model. This model was selected as the final candidate. Fig. 11 depicts its architecture; Figs. 12 and 13 illustrate respectively

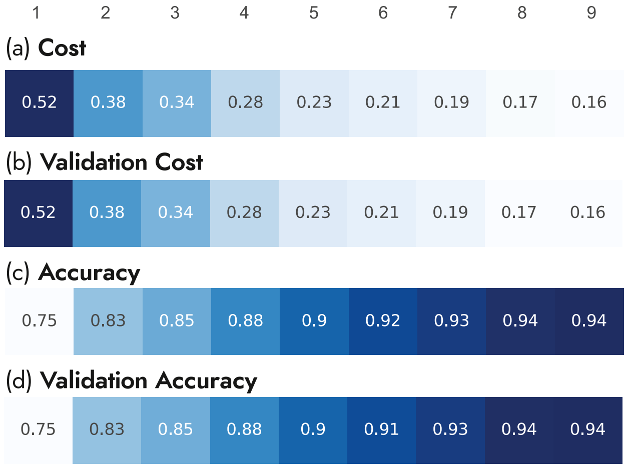
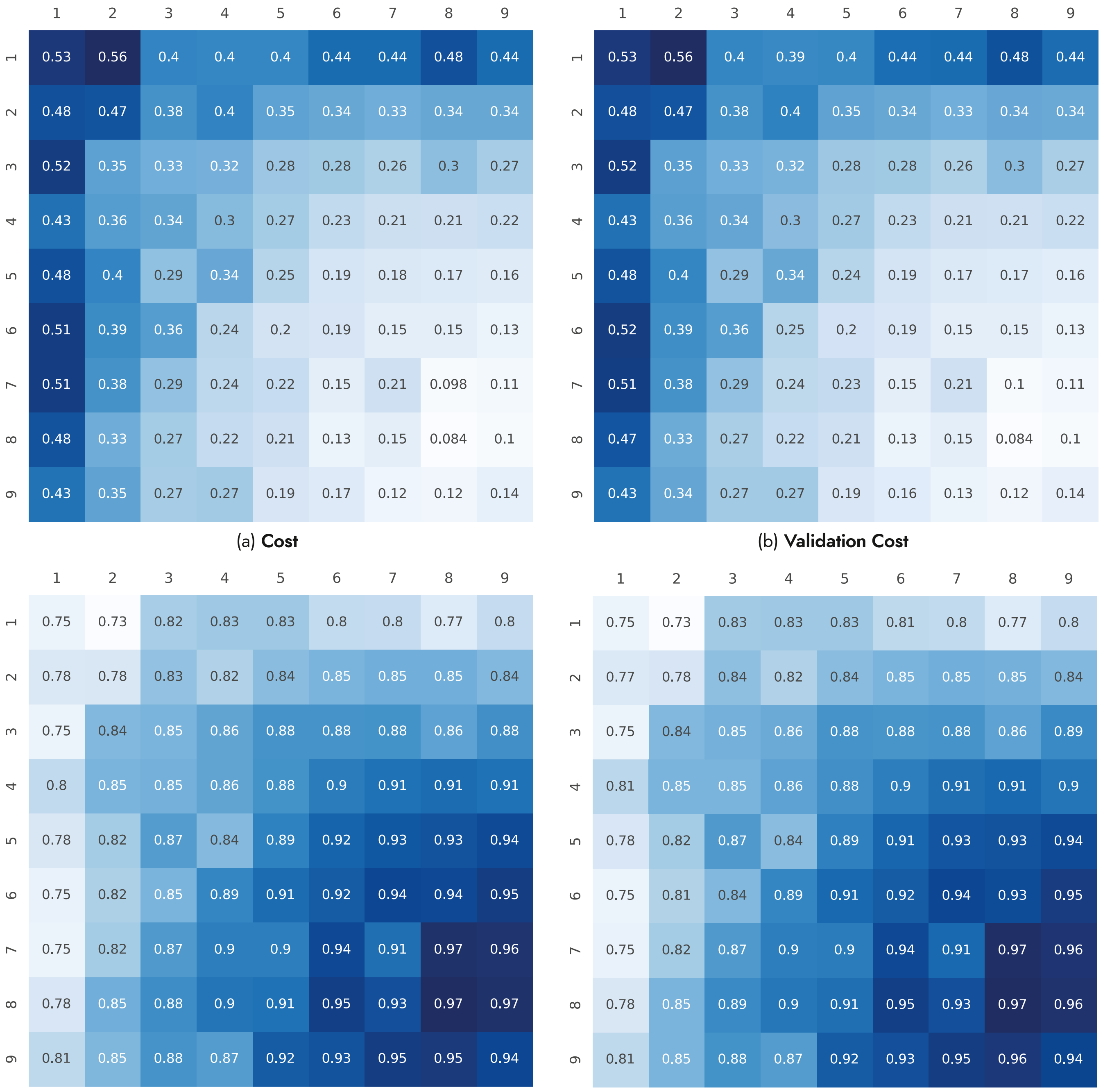


Fig. 9. Summary of the obtained (a) cost, (b) validation cost, (c) accuracy and (d) validation accuracy for the first group of models in function of the implemented neurons' number (upper horizontal axis within the hidden layer.



	1	2	3	4	5	6	7	8	9
1	0.53	0.56	0.4	0.4	0.4	0.44	0.44	0.48	0.44
2	0.48	0.47	0.38	0.4	0.35	0.34	0.33	0.34	0.34
3	0.52	0.35	0.33	0.32	0.28	0.28	0.26	0.3	0.27
4	0.43	0.36	0.34	0.3	0.27	0.23	0.21	0.21	0.22
5	0.48	0.4	0.29	0.34	0.25	0.19	0.18	0.17	0.16
6	0.51	0.39	0.36	0.24	0.2	0.19	0.15	0.15	0.13
7	0.51	0.38	0.29	0.24	0.22	0.15	0.21	0.098	0.11
8	0.48	0.33	0.27	0.22	0.21	0.13	0.15	0.084	0.1
9	0.43	0.35	0.27	0.27	0.19	0.17	0.12	0.12	0.14

(a) **Cost**

	1	2	3	4	5	6	7	8	9
1	0.53	0.56	0.4	0.39	0.4	0.44	0.44	0.48	0.44
2	0.48	0.47	0.38	0.4	0.35	0.34	0.33	0.34	0.34
3	0.52	0.35	0.33	0.32	0.28	0.28	0.26	0.3	0.27
4	0.43	0.36	0.34	0.3	0.27	0.23	0.21	0.21	0.22
5	0.48	0.4	0.29	0.34	0.24	0.19	0.17	0.17	0.16
6	0.52	0.39	0.36	0.25	0.2	0.19	0.15	0.15	0.13
7	0.51	0.38	0.29	0.24	0.23	0.15	0.21	0.1	0.11
8	0.47	0.33	0.27	0.22	0.21	0.13	0.15	0.084	0.1
9	0.43	0.34	0.27	0.27	0.19	0.16	0.13	0.12	0.14

(b) **Validation Cost**

	1	2	3	4	5	6	7	8	9
1	0.75	0.73	0.82	0.83	0.83	0.8	0.8	0.77	0.8
2	0.78	0.78	0.83	0.82	0.84	0.85	0.85	0.85	0.84
3	0.75	0.84	0.85	0.86	0.88	0.88	0.88	0.86	0.88
4	0.8	0.85	0.85	0.86	0.88	0.9	0.91	0.91	0.91
5	0.78	0.82	0.87	0.84	0.89	0.92	0.93	0.93	0.94
6	0.75	0.82	0.85	0.89	0.91	0.92	0.94	0.94	0.95
7	0.75	0.82	0.87	0.9	0.9	0.94	0.91	0.97	0.96
8	0.78	0.85	0.88	0.9	0.91	0.95	0.93	0.97	0.97
9	0.81	0.85	0.88	0.87	0.92	0.93	0.95	0.95	0.94

(c) **Accuracy**

	1	2	3	4	5	6	7	8	9
1	0.75	0.73	0.83	0.83	0.83	0.81	0.8	0.77	0.8
2	0.77	0.78	0.84	0.82	0.84	0.85	0.85	0.85	0.84
3	0.75	0.84	0.85	0.86	0.88	0.88	0.88	0.86	0.89
4	0.81	0.85	0.85	0.86	0.88	0.9	0.91	0.91	0.9
5	0.78	0.82	0.87	0.84	0.89	0.91	0.93	0.93	0.94
6	0.75	0.81	0.84	0.89	0.91	0.92	0.94	0.93	0.95
7	0.75	0.82	0.87	0.9	0.9	0.94	0.91	0.97	0.96
8	0.78	0.85	0.89	0.9	0.91	0.95	0.93	0.97	0.96
9	0.81	0.85	0.88	0.87	0.92	0.93	0.95	0.96	0.94

(d) **Validation Accuracy**

Fig. 10. Summary of the obtained (a) cost, (b) validation cost, (c) accuracy and (d) validation accuracy for the second group of models in function of the implemented neurons' number within hidden layer 2 (upper horizontal axis) and hidden layer 1 (left vertical axis).

the evolution of its mean cost and accuracy along the training/cross-validation process. The mean loss converges to a value of 0.16 while the mean accuracy to 0.97. No generalization gap is observed between the training and cross-validation curves which evolve tightly close to each other. This confirms that the model's learning has been performed within an optimal regime without underfitting nor overfitting the data. A further evaluation of model (7,8) on the test set yielded an accuracy of 0.96. By featuring only 5 neurons in the output layer, the model has a total number of 151 parameters. This number would have been 385 if a traditional labeling approach has been adopted. The latter would have required 31 output neurons to be implemented to handle the five-action control strategies.

6. Conclusion and recommendations

This work focused on developing a tinyML-oriented solution for a machine learning-based autonomous greenhouse microclimate control based on multi-variate sensed data (5 variables). The MLP architecture has been selected and various hyperparameters have been experimented while training 90 model instances. A robust five-fold training/cross-validation procedure on a balanced dataset was also carried out to find the optimal and best-performing model. From an evaluation metric-based perspective, multiple MLPs can fulfill the task; however, from a tinyML perspective, only the model with the less parameter number has to be kept for optimized computations and

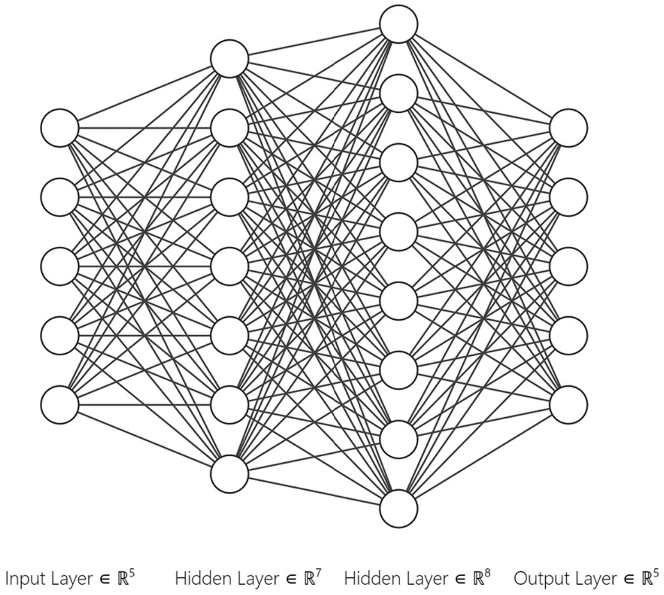


Fig. 11. Architecture of the final selected model (model (7,8)).

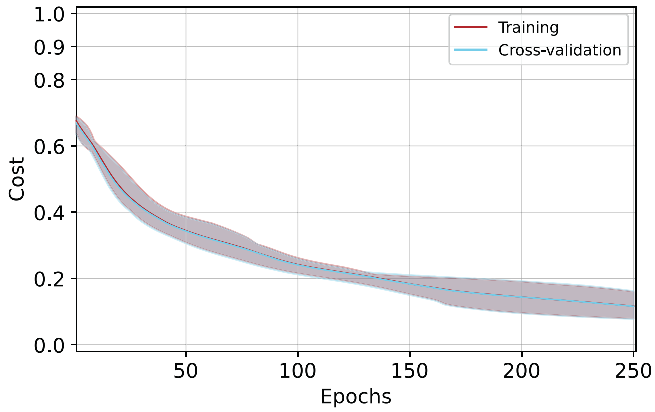


Fig. 12. The evolution of the mean cost along training/cross-validation for model (7,8).

minimum inference time. The selected model (model (7,8)) scored a mean accuracy of 97% on the validation sets, and 96% on our test set, with a number of 151 trainable parameters. Although the fact that the obtained model is a shallow neural network (which we consider as an advantage thanks to the adopted tinyML approach), it still can be categorized –from a broader definition perspective– under the deep learning field. The used multi-labeling approach enabled a parameter reduction of 60%. Thanks to its lightness, the model also eliminates the need for using any regularization technique to prevent overfitting the data.

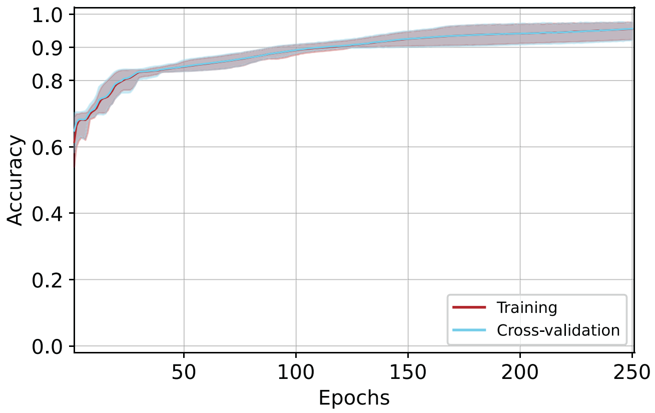


Fig. 13. The evolution of the mean accuracy along training/cross-validation for model (7,8).

While this study enabled to address multiple limitations of previous related works to the addressed thematic. There are several limitations along with future recommendations that we highlight:

- Our work has been conducted on one type of plants i.e., strawberry. Hence, we suggest:
  - o Conducting a review study of the various climatic variables that have to be monitored and taken into consideration for an autonomous greenhouse-based control of additional crop types.
  - o Unifying model development for the microclimate greenhouse control of multiple crop types.

- Although the obtained performance metric can be contended as satisfactory. We consider that it may be ameliorated further. Herein we suggest: (1) A model centric avenue, which focuses on modifying the parameters and hyperparameters; or a (2) Data-centric avenue which focuses on collecting additional and more refined data.

CRediT authorship contribution statement

Ilham Ihoume: Data curation, Formal analysis, Writing – original draft. **Rachid Tadili:** Supervision, Conceptualization, Methodology, Resources, Validation. **Nora Arbaoui:** Writing – review & editing. **Mohamed Benchrifa:** Writing – review & editing. **Ahmed Idrissi:** Writing – review & editing. **Mohamed Daoudi:** Writing – review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Alipio, M.I., et al., 2017. A smart hydroponics farming system using exact inference in Bayesian network. IEEE 6th Global Conference on Consumer Electronics (GCCE). IEEE, Nagoya, pp. 1–5.

Alwis, S.D., et al., 2022. A survey on smart farming data, applications and techniques. Comput. Ind. 138, 103624.

Alzubaidi, L., et al., 2021. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J. Big Data 8 (1), 53.

Angelopoulos, C.M., et al., 2020. Keeping data at the edge of smart irrigation networks: a case study in strawberry greenhouses. Comput. Netw. 167, 107039.

Apicella, A., et al., 2021. A survey on modern trainable activation functions. Neural Netw. 138, 14–32. https://doi.org/10.1016/j.neunet.2021.01.026.

Arias del Campo, F., et al., 2021. Auto-adaptive multilayer perceptron for univariate time series classification. Expert Syst. Appl. 181, 115147.

Azadi, H., et al., 2021. Rethinking resilient agriculture: from climate-smart agriculture to vulnerable-smart agriculture. J. Clean. Prod. 319, 128602.

Azaza, M., et al., 2016. Smart greenhouse fuzzy logic based control system enhanced with wireless data monitoring. ISA Trans. 61, 297–307. https://doi.org/10.1016/j.isatra.2015.12.006.

Cordovil, C.M., d. S., et al., 2020. Chapter 22 - climate-resilient and smart agricultural management tools to cope with climate change-induced soil quality decline. In: Prasad, M.N.V., Pietrzykowski, M. (Eds.), Climate Change and Soil Interactions. Elsevier, pp. 613–662.

Doshi, M., Varghese, A., 2022. Chapter 12 - Smart agriculture using renewable energy and AI-powered IoT. In: Abraham, A., et al. (Eds.), AI, Edge and IoT-based Smart Agriculture. Academic Press (Intelligent Data-Centric Systems), pp. 205–225.

Dutta, Dr.L., Bharali, S., 2021. TinyML Meets IoT: A Comprehensive Survey. Internet of Things 16, 100461. https://doi.org/10.1016/j.iot.2021.100461.

Escamilla-García, A., et al., 2020. Applications of artificial neural networks in greenhouse technology and overview for smart agriculture development. Appl. Sci. 10 (11), 3835.

Frankowiak, M., Grosvenor, R., Prickett, P., 2005. A review of the evolution of microcontroller-based machine and process monitoring. International Journal of

Machine Tools and Manufacture 45, 573–582. https://doi.org/10.1016/j.ijmachtools.2004.08.018.

Han, H., Siebert, J., 2022. TinyML: A systematic review and synthesis of existing research. 2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pp. 269–274.

Ihoume, I., et al., 2022. Performance study of a sustainable solar heating system based on a copper coil water to air heat exchanger for greenhouse heating. Sol. Energy 232, 128–138. https://doi.org/10.1016/j.solener.2021.12.064.

Iizumi, T., Ramankutty, N., 2015. How do weather and climate influence cropping area and intensity? Glob. Food Secur. 4.

Jamil, F., et al., 2022. Optimal smart contract for autonomous greenhouse environment based on IoT blockchain network in agriculture. Comput. Electron. Agric. 192, 106573.

Katzin, D., van Henten, E.J., van Mourik, S., 2022. Process-based greenhouse climate models: genealogy, current status, and future directions. Agric. Syst. 198, 103388.

Kingma, D.P., Ba, J., 2017. Adam: a method for stochastic optimization. arXiv, 1412.6980 1412.6980 [cs] [preprint]. Available at: http://arxiv.org/abs/1412.6980 (Accessed: 31 March 2022).

Lal, R., 2021. Climate change and agriculture. Climate Change. Elsevier, pp. 661–686 https://doi.org/10.1016/B978-0-12-821575-3.00031-1.

Liao, R., et al., 2021. Development of smart irrigation systems based on real-time soil moisture data in a greenhouse: proof of concept. Agric. Water Manag. 245, 106632.

Liu, D., Mishra, A.K., Ray, D.K., 2020. Sensitivity of global major crop yields to climate variables: a non-parametric elasticity analysis. Sci. Total Environ. 748, 141431.

Lopez-Ridaura, S., et al., 2021. Immediate impact of COVID-19 pandemic on farming systems in Central America and Mexico. Agric. Syst. 192, 103178.

Mani, V., Sankaranarayanan, S., 2017. IoT based smart energy management system. 12 (16), 8.

Mehra, M., et al., 2018. IoT based hydroponics system using deep neural networks. Comput. Electron. Agric. 155, 473–486.

Pedrycz, W., Chen, S.-M. (Eds.), 2020. Deep Learning: Concepts and Architectures. Springer International Publishing (Studies in Computational Intelligence), Cham.

Poudel, S., Shaw, R., 2016. The relationships between climate variability and crop yield in a mountainous environment: a case study in Lamjung District, Nepal. Climate 4 (1), 13.

Ray, P.P., 2022. A review on TinyML: state-of-the-art and prospects. J. King Saud Univ. Comp. Inform. Sci. 34 (4), 1595–1623.

Refaeilzadeh, P., Tang, L., Liu, H., 2009. Cross-validation. Encyclop. Database Syst. 532–538, 532–538.

Revathi, S., Sivakumaran, N., 2016. Fuzzy based temperature control of greenhouse. IFAC-PapersOnLine 49 (1), 549–554.

Sarker, I.H., 2021. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. SN Comp. Sci. 2 (6), 420.