Usage of Artificial Intelligence Systems and Working with the Neural Network in Assessing the Condition of Plants in Smart Greenhouses

Olha Danyltsiv, Andrii Khomiak, Oleg Nazarevych

Ternopil Ivan Puluj National Technical University, Ruska Street, 56, Ternopil, Ternopil region, 46001, Ukraine

Abstract

The paper proposes an approach to solving the problem of recognizing the state of plants using a neural network for use in information systems for monitoring and managing the microclimate for smart growing boxes and smart greenhouses. The developed smart greenhouse prototype is presented, which includes information technology for monitoring temperature, humidity, soil and recording photos of plants and further analysis of plant condition using a neural network (deeplearning). In addition, a telegram-bot interface is built for easier control light, temperature and other factors that create a favourable microclimate for a particular type of plants and manage through smartphone.

The main purpose of the work is to adapt the algorithm for training an artificial neural network using a GPU and the ability to assess the condition of plants based on photos for use in information technology control of smart greenhouse prototype.

Keywords 1

Artificial intelligence, smart greenhouse, deeplearning, plant physiology

1. Introduction

In recent decades, there has been an increased interest in neural networks in connection with the implementation of the SMART concept in various spheres of human life. Neural networks started to be explored and introduced into everyday life not only by specialists in technology, physiology and psychology but also by ordinary users. This is logical, because the artificial neural network, in fact, is a model of the natural nervous system of a living organism, so the creation and study of such networks allows us to learn more about the functioning of natural systems [1].

An important factor and a unique feature of the process is that the neural network draws certain generalized conclusions automatically. Due to its structure, it is able to analyze, compare, process information without requiring any specific programs. Another property of neural networks is reliability. Even when some elements of the network work incorrectly or simply fail, the network is still able to give the correct results, albeit with less accuracy.

There are special types of neural networks, they are able to generate an abstract image, which is based on input signals. As an example, a network is a sequence of distorted images of a symbol, letter, or punctuation mark. After training, the network will be able to generate this input without distortions.

This indicates that the network is able to generate unique conclusions based on the input information obtained during training using the finished dataset.

Thus, today the study of neural network training methods is a really relevant topic that can be applied to the concept of SMART greenhouse management.

MoMLeT+DS 2021: 3rd International Workshop on Modern Machine Learning Technologies and Data Science, June 5, 2021, Lviv-Shatsk, Ukraine

EMAIL: olhadanyltsiv@gmail.com (O. Danyltsiv); andrewhamster@hotmail.com (A. Khomiak); taltek.te@gmail.com (O. Nazarevych)
ORCID: 0000-0002-2014-0997 (O. Danyltsiv); 0000-0002-1763-868X (A. Khomiak); 0000-0002-8883-6157 (O. Nazarevych)

The main purpose of the work is to adapt the algorithm for training an artificial neural network using a GPU and the ability to assess the condition of plants based on photos for use in information technology control of smart greenhouse prototype.

In our research we distinguish the following tasks:

1. Choose the type of neural network for the classification and assessment of plants.
2. Form, organize, filter and systematize a dataset for network research (training) with the help of smart greenhouse management information technology.
3. Carry out the marking of the condition of a plant on the basis of the formed dataset.
4. Implement neural network training using GPU.
5. Evaluate the results obtained.

2. The concept of artificial neural network

Since the 1980s, the interest in artificial neural networks grew rapidly. Specialists in various fields of technical design, philosophy, physiology and psychology were interested in the opportunities provided by this technology. They actively sought to apply it in their fields within development projects and research.

In 2007, Jeffrey Hinton at the University of Toronto developed algorithms for deep learning of multilayer neural networks. This event marked the beginning of the active use of artificial neural networks in various spheres of human life. In turn, this gave an impetus to the development of artificial intelligence technology [1].

Artificial neural networks (ARNs) are among the most common classification methods in data mining. They are also a powerful method of forecasting. Such systems, as a rule, learn to solve problems, considering examples in general, i.e. the research is perceived integrally, without any specific programming or any prior knowledge about it.

Two strengths of artificial neural networks are the parallelization of information processing and the ability to self-learn, i.e. to create abstractions.

What is an abstraction? Abstractions are the ability to output an acceptable result based on the data used in the learning process. Such features allow neural networks to solve large-scale problems that are still considered difficult to solve. Despite this, artificial neural networks cannot provide ready-made solutions directly; they need to be integrated into more complex systems instead. For example, complex problems can be divided into simpler ones, which will be solved by neural networks [2].

The main features of artificial neural networks are:

1. The obviousness of the answer. For example, if we consider the classification problem, the neural network can be designed to provide information not only about the class of input data but also about the reliability of the result. Such information will provide an opportunity not to take into account questionable decisions of the neural network [1].
2. Nonlinearity. Because the activation function of each neuron is often chosen to be nonlinear, the network as a whole can mimic much more complex nonlinear functions. This allows the ANN to approximate functions of almost any complexity. The only limitation is the number of neurons and the structure of the network connections.
3. Clear correspondence of the input information to the output.
4. Adaptability. Neural networks learn to flexibly solve problems. In particular, they can be retrained quite easily if the fluctuations in environmental parameters are insignificant. Also, high adaptability allows the reliable use of neural networks in non-stationary environments. However, it should be noted that high adaptability can lead to unstable behaviour of ANN. If it quickly adapts to certain stimuli when changing the coefficients, it can significantly affect further performance. That is, high adaptability can lead to loss of stability and vice versa. This is usually called the problem of stability-plasticity.
5. Fault tolerance. A well-trained neural network can function properly, even if several of its neurons fail. This would allow for the implementation of neural networks on a physical level and confidence in the reliability of the chosen architecture. Because the information needed for the network to function is distributed in each neuron, the entire network can fail only if its structure is severely

damaged, while minor damage will never seriously affect performance. This is an obvious advantage of ANN.

6. Ability to scale. The architecture of neural networks makes it very easy to perform a large percentage of calculations simultaneously. This allows a minimal amount of effort to transfer or scale systems based on artificial neural networks.

3. Types of neural networks

Neural networks are divided into single-layer and multilayer ones [3].

The set of neurons combined in one layer is called a single-layer neural network. A multilayer neural network is formed when the combination of single-layer neural networks into a single entity takes place [4].

Multilayer networks between input and output data have several layers of hidden neurons. These layers add more nonlinear connections to the model. They provide greater accuracy in the process. For example, a multilayer perceptron with sigmoid activation functions is able to approximate an arbitrary functional dependence.

It should be noted that an important advantage of neural networks over classical means of classification and prediction is their ability to learn [4].

The very phrase "teach the neural network" means to show it what exactly is expected of it in action. This process can be compared to teaching numbers to a child. After showing a picture with the image of the number "5" to a child, we later ask: "What is this number?". In that case, if the answer is incorrect, we tell the child the correct answer, i.e.: "This is the number five". The child remembers this example together with the correct answer, i.e. in their memory, there are certain changes in the right direction. Then we will repeat the process of presenting the numbers again and again until everything is studied.

When teaching a neural network, similar actions are performed. That is, we have a database that contains materials for training (in the case of this research the materials are graphical).

Presenting the image to the input system of the neural network, we get a certain answer from it.

It is important to note that after repeated presentation of the examples, the weights of the neural network are stabilized and it gives the correct answers to all (or almost all) examples from the database. In this case, it is said that "the neural network has studied all the examples", "the neural network has been taught", or "the neural network has been trained". In software implementations, we can see that during the learning process the magnitude of the error (the sum of the squares of errors for all outputs) gradually decreases. When the margin of error reaches zero or an acceptable low level, the training is stopped, and the resulting neural network is considered trained and ready for use on new data.

All the information that the neural network has about the task is contained in a set of examples. Therefore, the quality of neural network learning directly depends, first of all, on the number of samples in the training database, as well as on how fully and holistically these samples describe this task. That is, for example, it makes absolutely no sense to use a neural network to predict the financial crisis if the training sample for crisis prediction is not presented. It is believed that for a full-fledged training of the neural network one would need at least a few dozen (preferably hundreds of) examples.

It is worth mentioning that teaching a neural network is a rather complex and knowledge-intensive process. Learning algorithms for such systems have different parameters and settings, and to manage them one has to have an understanding of their impact.

4. Types of training

As mentioned in the previous paragraph, in order for a network to solve the tasks set before it, it must first be trained. This ability is considered the main property of the brain. Next, the relationship between the type of training and its effectiveness has to be considered. The learning process for such systems involves adjusting the structure of connections between individual neurons and synoptic connections that affect the signals of the coefficients. Such complexes allow the neural networks to solve the tasks effectively. Basically, neural network learning is based on a sample set (data set). Scientists and researchers in this field have developed special algorithms for teaching neural networks.

Such algorithms allow to increase the efficiency of response to input signals and expand the scope of their application.

Examples of these are the so-called Hebb training (Donald Gebb, who hypothesized training based on the neuroplasticity), Rosenblatt (he is considered the founder of the perceptron), Alexei Ivakhnenko and Valentin Lapa (researchers of the first working multilayer neural networks). Such algorithms allow to increase the efficiency of response to input signals and expand the scope of their application.

A large number of different technologies are used for machine learning. In particular, discriminant analysis, Bayesian classifiers and numerous other mathematical methods can be used. But at the end of the 20th century, artificial neural networks (ANN) started receiving more and more attention. Now they are not only used in various spheres of economic and social activities, but are increasingly used by ordinary people. One of the largest bursts of interest in them began in 1986, after the significant development of the so-called "method of backpropagation", which was successfully used in the training of the neural network.

ANN is a system of connected and interacting artificial neurons, built on the basis of relatively simple processors. Each ANN processor periodically receives signals from one processor (or from sensors or from other signal sources) and periodically sends signals to other processors. These simple processors, which are arranged into a network, are able to solve quite complex problems.

Most often, neurons are arranged in the network by levels (which are more often called layers). Let us have a look at a multilevel neural network. First-level neurons are usually input neurons. They receive data from the outside (for example, from the sensors of a particular recognition system) and after processing transmit pulses through the synapses of neurons at the next level. Neurons at the second level (this level is called hidden because it is not directly connected to either the input or output of the ANN) process the received pulses and transmit them to the neurons at the output level. The simulation of neurons is involved in the process, hence each input-level processor is associated with several hidden-level processors, each of which, in turn, is associated with several output-level processors. This is the architecture of the simplest ANN, which is capable of learning and can find simple connections in the data.

Deep learning is applied only to more complex artificial neural networks containing several hidden levels. At the same time levels of neurons can alternate with layers that carry out difficult logical transformations or calculations. Each subsequent layer of the network looks for connections in the previous one. An ANN arranged in this way is able to find not only simple connections but also links between connections. Google is a fine example of employing a similar system. After the transition to a neural network with the in-depth learning, the company has managed to dramatically improve the quality of its popular product Google Translate.

To summarize, there are three types of artificial neural network training: with a teacher, without a teacher, and mixed.

ANN training with a teacher is one of the most popular ways. It basically contains a change in weight coefficients using training examples. For each input example, there is a corresponding output vector. In the process of training, an example is selected randomly, and the neural network forms the original vector based on the example. This vector is compared with the desired one and according to a special algorithm, the process of modification of weighting factors takes place in such a way as to approach the desired vector [4].

When learning without a teacher, the model has a data set without explicit instructions on what to do with it. The neural network tries to find connections in the data on its own, extracting useful features and analyzing them.

Depending on the task, the model organizes the data differently:

- Clustering. Even without special knowledge, you can look at images of animals and divide them into groups by species, simply based on the shape, size or presence of fur, and so on.
- This is called clustering – the most common task for learning without a teacher. The algorithm selects similar data, finds common features, and arranges them into groups.
- Detection of anomalies. Banks can detect fraudulent transactions by detecting unusual actions in the purchasing behaviour of customers. For example, it is suspicious if one credit card is used in two different countries or on different continents on the same day.

- Associations. Considering several key features of the object, the model can predict other cases where a connection is present [1].

5. Description of the research prototype

Passing in the optimizer and loss function is optional, and in many situations fastai can automatically select appropriate defaults.

Learner is also responsible (along with Optimizer) for handling fastai's transfer learning functionality. When creating a Learner the user can pass a splitter. This is a function that describes how to split the layers of a model into PyTorch parameter groups, which can then be frozen, trained with different learning rates, or more generally handled differently by an optimizer.

One area that we have found particularly sensitive in transfer learning is the handling of batch-normalization layers [3]. We tried a wide variety of approaches to training and updating the moving average statistics of those layers, and different configurations could often change the error rate by as much as 300%. There was only one approach that consistently worked well across all datasets that we tried, which is to never freeze batch-normalization layers, and never turn off the updating of their moving average statistics. Therefore, by default, Learner will bypass batch-normalization layers when a user asks to freeze some parameter groups. Users often report that this one minor tweak dramatically improves their model accuracy and is not something that is found in any other libraries that we are aware of.

DataLoaders and Learner also work together to ensure that model weights and input data are all on the same device. This makes working with GPUs significantly more straightforward and makes it easy to switch from CPU to GPU as needed.

Here the developed information part is described. It consists of many levels (such as physical, network and software levels) and provides the collection, retrieval, processing and transmission of information. Such systems allow solving a wide range of tasks of different nature, including organizational tasks [5].

The advantages of "smart" greenhouses, in general, are quite obvious. This is not just a place where plants are completely protected from external negative factors, but can also bear fruit all year round with proper care.

Due to the automation of routine processes, many trivial tasks can be performed without human input.

Of course, not just any "smart" greenhouses are able to grow plants without human intervention, but their capabilities are quite wide.

Such greenhouses are able to maintain a comfortable temperature inside. The room is usually ventilated automatically, so on hot days vegetables, fruits, berries, etc. do not wither in nature-dictated conditions.

- Water the plants at the specified time. The irrigation system is usually automated and extremely easy to operate. This is so that any user, including a novice gardener or a person unfamiliar with the principles of the system, can easily figure the setup out.

The system is also capable of providing automated control of processes, which is of interest to modern science and humanity as a whole. This process is the cultivation of plants that can have different purposes (decorative or for cooking, main course or seasoning). Different species and different environments require different approaches to this process [5].

Smart Growing Box (SGB) is a concept of a physical device for seed germination/plant cultivation, which automatically maintains an ideal environment for a particular plant (lighting provided for the specific part of the day, temperature and humidity or microclimate) [7].

Inside the SGB there is a lamp, a fan and a heater that switches on automatically to maintain a microclimate set according to cloud service recommendations based on SGB location data, the type of plant grown in it and analytical data collected from similar SGBs. Data is exchanged using the MQTT protocol, which is based on the TCP transport layer protocol and implements the publish-subscribe model. Its advantage is low redundancy, which is critical for embedded devices with low power and limited network bandwidth. In this information system, the cloud is a set of services available via the

Internet that provide reception, storage and processing of data received under the MQTT protocol. Cloud data is stored in two types of databases:

- SQL for storing static user data, SGB settings.
- Elasticsearchserver [6] for storage and further machine learning based on analytical data obtained from sensors. The cloud will also have its own ApplicationProgramInterface (API) to interact with the web application, which will duplicate the functionality of the telegram bot and additionally have the visualization of analytical data in the form of real-time graphs within 1 minute.

An intelligent greenhouse that uses the principles of artificial intelligence as a key principle of operation is a design in which the main processes are automated using a model of neural connections, computer vision and appropriate plant growth sensors.

Such constructions are aimed at facilitating the cultivation of plants, reducing and optimizing the time of their care, providing and maintaining a favourable microclimate with mandatory analysis of humidity, temperature, ventilation and more.

So, lets describe standard example of how to fine-tune an ImageNet [17] model on the Oxford IIT Pets dataset [18] and achieve close to state-of-the-art accuracy within a couple of minutes of training on a single GPU.

At first we should imports all the necessary pieces from the library. fastai is designed to be usable in a read–eval–print loop (REPL) environment as well as in a complex software system. Then we connect with standard dataset.

This is an abstraction that represents a combination of training and validation data and will be described more in a later section. The next step is fitting the model. In this case, it is using the 1cycle policy [19], which is a recent best practice for training and is not widely available in most deep learning libraries by default. It is annealing both the learning rates, and the momentums, printing metrics on the validation set, displaying results for example in an HTML table (if run in a Jupyter Notebook, or a console table otherwise), recording losses and metrics after every batch to allow plotting later, and so forth. A GPU will be used if one is available.

In modern natural language processing (NLP), perhaps the most important approach to building models is through fine-tuning pre-trained language models.

Projects like this typically use climate controllers, artificial irrigation systems, ventilation, air irrigation and plant or emergency warning systems. Most smart greenhouse solutions focus on the proper operation of automatic control systems, which allow you to program care actions according to time or depending on the performance of the system sensors.

Modern technology makes it possible to develop smart greenhouses on the basis of a popular platform for IoT (Internet of things) - ESP, which is usually controlled from a mobile device or via the Internet.

Let us look at the principle of the Internet of things in more detail. The very idea that devices can exchange information with each other without human intervention, appeared a long time ago.

The modern "Internet of things" involves network communication of objects that interact with each other and the environment. You can control a variety of devices with your phone from anywhere. However, having a stable uninterrupted internet connection is a requirement. In addition, interconnected devices can perform certain actions without user intervention.

Among the important functions of the Internet of things are the facilitation of everyday life, improving efficiency and quality of work and energy preservation. In general, we can state that there is no clear list of devices for which this approach can be applied. That is, this idea can be implemented in household appliances: a remote control washing machine, or a refrigerator capable of compiling shopping lists and ordering home delivery.

Another option is well-known gadgets that can be worn. There is probably no person who has not heard of fitness trackers and smart watches. IoT also includes cars or other vehicles with an autopilot system (meaning those that can drive without a driver behind the wheel).

However, it is important to note that the Internet of Things is not limited to "smart" appliances. It can be used in any industry where certain processes can be automated. This means it can be useful in almost all spheres of human activity and life. IoT is especially active in the agricultural sector, logistics, and the concept of Smart City. This is useful for cases where there is a need to monitor the condition of objects or to collect large amounts of data for further analysis.

Based on the above, it is logical that the idea of IoT is used in this paper.

A prototype of a smart greenhouse prototype was developed to implement this information technology of monitoring and control. 7.8 As a result of such realization, we have an opportunity to receive data on air and soil temperature and relative humidity. Record the modes of operation of actuators for air ventilation, drip irrigation and heating.

To enable the accumulation of current images of plants in this information technology, webcams and a server for storing photos (Raspberry Pi3) were installed. [8]

The concept of monitoring and control of such a smart greenhouse prototype is shown in the figure below.

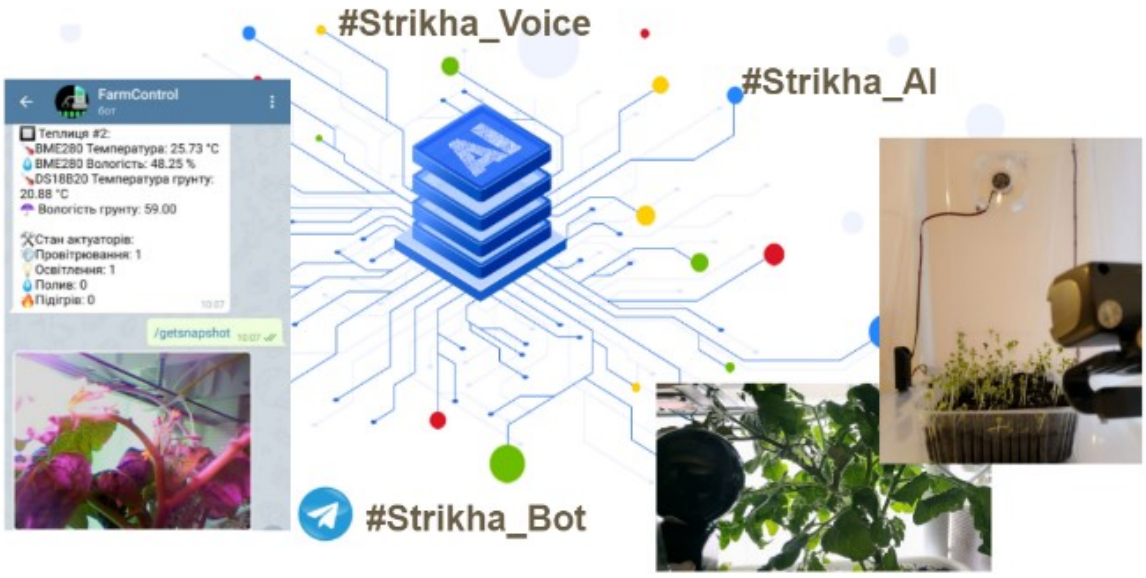

Figure 1: The concept of monitoring and management of a smart greenhouse prototype

The general concept of the prototype of such a smart greenhouse prototype can be seen in the video [9].

It is based on the concept of a device for growing different types of plants by supporting automatic watering, light control, temperature and other factors that create a favourable microclimate for a particular type of seed [8].

The prototype itself contains temperature sensors, automated ventilation and heating, based on pre-collected data on a particular plant. An important element of a smart greenhouse is also watering, set for certain hours or days. The collected indicators form four main graphs, which are presented below in Figure 2.

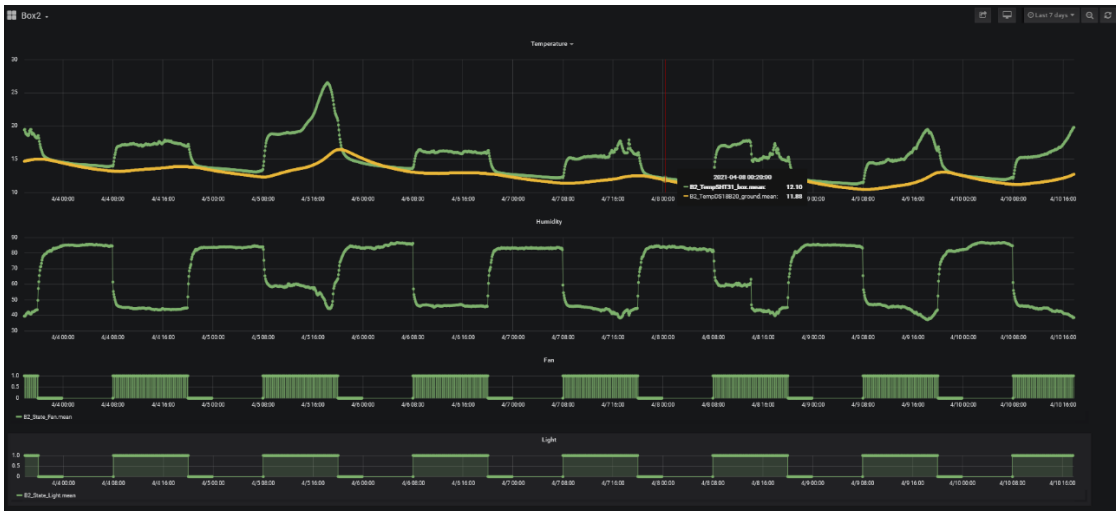

Figure 2: Graphs of temperature, humidity, ventilation and lighting of the system

To display graphs of monitoring indicators of smart greenhouse prototype the Grafana system, which is hosted on the management server of smart greenhouse prototype, was used [10].

The greenhouse is equipped with cameras that take pictures of plants at set time intervals in order to further analyze their condition and make clear recommendations for care. These actions occur through the use of artificial intelligence neural networks.

6. Description of the software algorithm

In any process of work, development or research stages are usually outlined. When the project is divided into clearly defined tasks, the implementation is more convenient and issue-free.

6.1. Description of the dataset structure

Neural network learning begins with the preparation of materials. These materials are usually marked images of objects, in our case – photos that will need to be recognized. It is with their help that the neural network is trained. In fact, in this way we teach the NN to analyze what exactly is depicted in the photo and the condition of the object depicted. However, in order to have this "training data" in the form required, it is necessary to form such a database.

The so-called "raw" data is a large sample of photos taken at intervals using a webcam in the developed box. What it looks like at the beginning is shown in Figure 3.

Input data encompasses the names of the photos, the photos themselves that contain some info about the plant, the stage of its growth, as well as water and temperature levels. The data also includes temperature conditions in the room where the experiment is performed, the temperature outdoors, the temperature of the ground for plants and the temperature in the experimental box with plants.

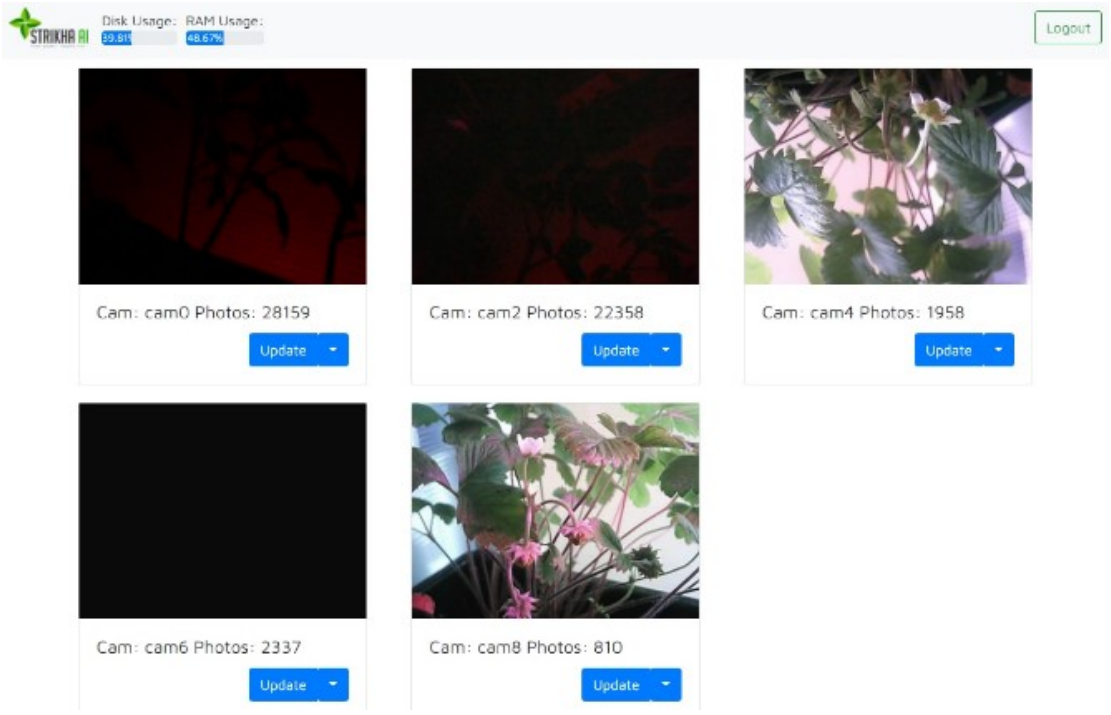

Figure 3: Learning objects for database formation

Brief characteristics of the plant are the data given as to whether it is well lit, watered sufficiently, whether the leaves stretch upwards or wither, etc. Some of them are given below:

- • Temp_outside.
- • Temp_room.
- • Temp_box.

- • Temp_ground.
- • Is_fito_lamp_on.

In our case, the authors do not follow strict terms to describe the condition of plants in terms of plant physiology. The authors have formed a simplified list of markers of plant condition, as the goal is to focus on the adaptation of techniques and training of the neural network.

6.2. Neural network training

The following tools are used to implement the training process:

- Anaconda is a common Python platform for data processing. The platform includes a collection of about 200 open-source packages and Intel MKL optimization. The platform specializes in "scientific computing": the science that deals with data, application of machine learning methods, large-scale info processing, forecasting etc. The use of the platform provides for simplified management of packages and their deployment [11];
- Anaconda Navigator is a desktop user interface that is part of the Anaconda distribution and allows users to run related programs and manage packages, among others.
- Conda is a freely and openly distributed cross-platform and language-agnostic package manager and Anaconda environment management system that installs and updates the packages and their dependencies. Originally designed for Python programs, today it can package and distribute software for a very wide range of languages.
- Python is one of the most powerful and widely used programming languages, which is quite easy to master without special skills. It has efficient high-level data structures and a simple but effective approach in terms of object-oriented programming [12].

It is also important to note that the list of existing Python libraries provides a wide range of tools for solving problems related to the creation of artificial intelligence and data analysis. With their help, engineers and scientists can cope with the tasks of image analysis, text and visualize the analyzed data much easier.

Python is also successfully used to implement Geographic Information Systems (GIS): cartography, geolocation, and GIS programming.

Typically, the advantage of Python over specialized languages created by scientists for scientists is that it is much easier for programmers to do more scientific tasks without learning a separate language each time. Also, using the Python language, the user will be able to more easily integrate the solution of a scientific problem with the practical goals of the project itself. Therefore, if there is a need for scientific calculations or data analysis one should consider this programming language.

During program development, the main packages used are:

- Pandas is one of the Python libraries that works with tabular data and has the technology to optimally process it. The library introduces two basic concepts that describe the data structures in it:DataFrame is a data object that is best represented in the form of a regular table because it is a tabular data structure. Series is an object similar to a one-dimensional array (an example is a list in Python), but the difference is the presence of associated labels, or so-called indexes, along with each element in a list. This feature turns it into an associated Python array or dictionary.
- The Fast.AI library is a top-level software add-on to the Pytorch package for the Python programming language. All these constructs are designed to facilitate the creation of software models of deep neural networks. Fast library. AI uses the best practices and approaches. At the moment, it has a wide range of opportunities [13]. FastAi is organized around two main design goals: to be approachable and rapidly productive, [15]. We wanted to get the clarity and development speed of Keras [16] and the customizability of PyTorch. This goal of getting the best of both make us to design of a layered architecture.
- NumPy is a Python language library for working with multidimensional data arrays. Typically, these arrays are indexed by positive integers. The data that the library works with is homogeneous. So what does that mean? Homogeneity of data allows optimizing work in comparison with standard lists of language [14].

- Scikit_Learn - Another library of machine learning software for the Python programming language. It is closely related to the previous library, as it is Scikit_Learn that mainly takes NumPy arrays as input.

There is a number of different work teams that were used in our study. For example, the pd.read_excel command is used to read a markup file. Then file names are converted to the file path on a disk.

Later, we filter the files, because only those that are on the disk are needed for work.

Let's move on to the method that trains the model for this target variable (label_name) based on the pre-trained ResNet34 architecture (ResNet is an abbreviated name for Residual Network (see Figure 4).

Let us move on to ResNet, also known as a residual neural network. In short, it is an artificial neural network (ANN), based on structures known from pyramidal cells of the cerebral cortex.

Residual neural networks do this through bandwidth connections or shortcuts to go through some layers. Typical ResNet models are implemented with two-layer or three-layer passes. They contain nonlinearities and batch normalization between them.

```
tfms = get_transforms()
```

```
def train_model_for_label(label_name):
    print(f"Selecting features for {label_name} classifier...")
    df[label_name] = df.labels.apply(partial(get_label_value,␣
↪label_name=label_name))
    data = df[['image_path', label_name]]
    print(data.head())
    print(data[label_name].value_counts())
    print(f"Creating databunch...")
    bunch = ImageDataBunch.from_df(Path("./images/"), data, bs=BATCH_SIZE,␣
↪valid_pct=0.05, ds_tfms=tfms)
    bunch.show_batch(ds_type=DatasetType.Valid)
    print(f"Starting training...")
    model = cnn_learner(bunch, base_arch=models.resnet34, metrics=accuracy)
    model.fit_one_cycle(4)
    model_file = f"clf_{label_name}_0"
    model.save(model_file)
    print(f"Saved weights in {model_file}")
    interpretation = model.interpret()
    interpretation.plot_confusion_matrix(4)
```

Figure 4: Syntax of the model training method

With the command train_model_for_label ("label_name") you can see the work of the training algorithm to mark the state of the leaves of the plant. Such training is conducted for all available labels.

6.3. Results

After completing the process of training the model, we obtain the so-called table result, which is shown in Figure 5.

Here you can find information about the stages of neural network learning, training losses, actual losses, the accuracy of training and time spent on each stage.

epoch	train_loss	valid_loss	accuracy	time
0	0.577004	0.066567	0.963415	00:56
1	0.281548	0.100566	0.975610	00:54
2	0.178278	0.013852	1.000000	00:54
3	0.131060	0.010825	1.000000	00:55

```
Saved weights in clf_light_0
```

Figure 5: Initial table of model training results

Confusion matrix is also presented as a source element. Its construction is provided by the Sklearn library commands.

The main purpose of using the confusion matrix is to assess the quality of the classifier output on the data set. That does, in fact, show how accurate our model is. Diagonal elements represent the number of points for which the expected label is equal to the real label, and outside the diagonal elements – those that are marked by the classifier incorrectly. The higher the value of the diagonal numbers of the confusion matrix, the better (the values themselves range from 0 to 1), as this indicates a large number of correct predictions.

Figure 6 shows the confusion matrix for training the lighting threshold.

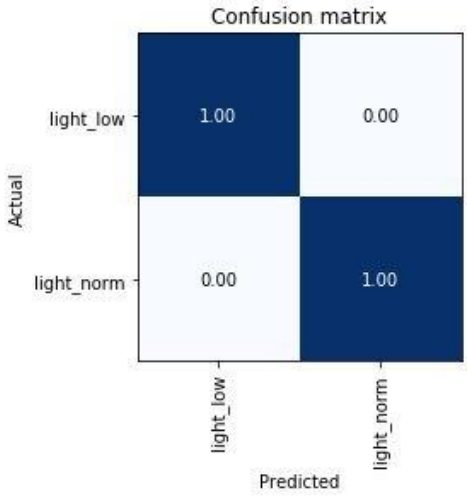

Figure 6: Training quality matrix for the illumination column model

Analyzing the matrix in relation to the above, we see that since the diagonal elements are equal to ones and the lateral elements to zeros, the quality of the neural network is satisfactory, and the training has been conducted correctly.

We shall present the same matrices for some other characteristics. For example, the matrix for the label "leaf position" is shown in Figure 7, and for the values of "humidity" – in Figure 8.

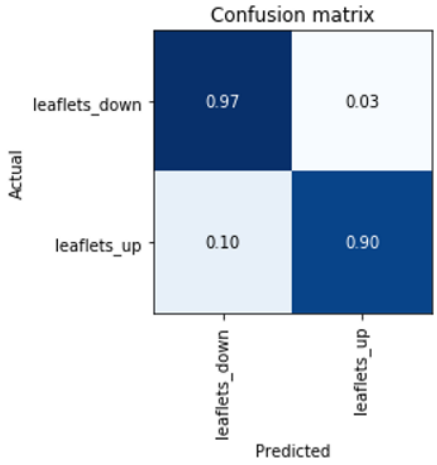

Figure 7: Training quality matrix model for the position of plant leaves

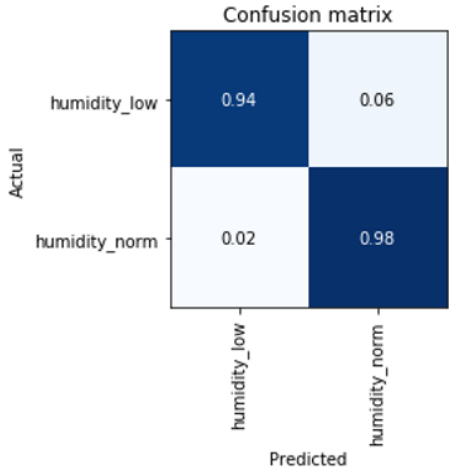

Figure 8: Training quality matrix model for humidity values

If the matrices on the diagonals are different from the unit value but quite close, this is similarly an indicator of the high quality of the algorithm.

It should be noted that the algorithm itself shows the method of image classification. That is, if you focus, for example, on the characteristics of lighting, it highlights the most contrasting images in this regard and divides them into certain groups. This ensures better network performance in the future.

6. Conclusions

A result of the constructed prototype of the smart greenhouse prototype and using information technology to monitor and manage the prototype, a set of observation data for the period of plant growth (within 20 days) was formed.

The neural network was adapted for the studied dataset, machine learning was performed using the deeplearning network and the corresponding reliable learning results were obtained, as evidenced by the training quality matrices.

As a result of the training, the available information technology of monitoring and management of a smart greenhouse prototype and an opportunity to receive a condition of plants on the basis of a photo from webcams placed in it was improved.

Regarding the further development of this project, the need for more rigorous formalization of the list of markers in accordance with the requirements of plant physiology should be noted. After making such changes to the list of markers for the formation of a new dataset and retraining of the neural network.

In this study, we have discussed the advancements of CNN in image classification tasks in general too. We have experienced that combining inception module and residual blocks with conventional CNN model, and ResNet gained better accuracy than stacking the same building blocks again and again.

In fact, the proposed prototype has two directions of further development: for smart growing box, which can be placed in a house or apartment (for growing greenery or seedlings) and for smart greenhouse which are usually placed on land. In both cases, it will be important to implement this system for assessing the condition of the plant on the base of the proposed approach to the construction of a neural network.

7. References

[1] X. B. Olabe, Redes Neuralna Artificiales y sus Applicaciones Format Impreso: Publicaciones de la Escuela de Ingenieros, 1998.

[2] M. Minsky, S. Papert, Perceptrons, Mir, Moscow, 1971.

[3] D.V. Efimov, V.A. Terekhov, I.Yu. Tyukin, Neural network control systems: Textbook. Manual for universities. Moscow, 2002.

[4] P.M. Hrytsiuk Application of artificial neural networks for long-term forecasting of grain yield 2018.

[5] O. Nazarevych, A. Volokha, O. Zymnytsky. Multilevel information system of ecomonitoring and climate control Smart Growing box, Innovative technologies: Materials of XVI scientific and technical. conf. students, graduate students, doctoral students and young scientists, INTL NAU, Kyiv, 2019.

[6] R. Kuc, M. Rogozinsk, Mastering Elasticsearch, Second Edition, 2015.

[7] F. Xia, L. Yang, L. Wang, A. Vinel, Internet of Things, International Journal of Communication Systems, no. 9, vol. 25, Wiley, 2012. doi.org/10.1002/dac.2417

[8] O.B. Nazarevych, V. Melnyk, Information technology of eco-monitoring of the farm for germination of microgreens based on the Raspberry Pi3 platform using, Proceedings of the scientific conference of Ternopil National Technical University named after Ivan Pulyuy, Ternopil, 2019.

[9] StrikhaAI DD SLIDESHOW, 2019. URL: https://youtu.be/NYeijedEHA4.

[10] Grafana: The open observability platform. Grafana Labs. URL: https://grafana.com/.

[11] A. Matthes, J. Tacke, Python Programming: 3 Menuscripts Crash Course Coding With Python Data Science, 2020.

[12] J. Joshua Thomas, P. Karagoz,, B. Bazeer Ahamed, P Vasant, Deep Learning Techniques and Optimization Strategies in Big Data Analytics, 2020.

[13] A. Anisimova, Y. Doroshenko, S. Pogoriliy, Y. Dorogy, Programming of numerical methods in Python language, Kyiv, Kyiv University Publishing and Printing Center, 2014.

[14] L. Chin, T. Dutta, NumPy Essentials, 2016.

[15] J. Howard, S. Gugger, Deep Learning for Coders with fastai and PyTorch: AI Applications Without a PhD, 1st ed., O'Reilly Media, Inc., 2020.

[16] F. Chollet, others, Keras, 2015. URL: https://keras.io.

[17] J. Deng, W. Dong, R. Socher, ImageNet: A Large-Scale Hierarchical Image Database, CVPR09, 2009. doi:10.1109/CVPR.2009.5206848

[18] O. Parkhi, A. Vedaldi, A. Zisserman, C. Jawahar, Cats and Dogs, IEEE Conference on Computer Vision and Pattern Recognition, 2012. doi: 10.1109/CVPR.2012.6248092.

[19] L. Smith, A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay, CoRR 2018. doi:abs/1803.09820.

[20] Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. CoRR 2015. doi:abs/1502.03167.