Agris on-line Papers in Economics and Informatics

Volume XIII Number 1, 2021

A Strategic Analytics Using Convolutional Neural Networks for Weed Identification in Sugar Beet Fields

Brahim Jabir, Noureddine Falih, Asmaa Sarih, Adil Tannouche

Faculty of Sciences and Technics, Sultan Moulay Slimane University, Beni Mellal, Morocco

Abstract

Researchers in precision agriculture regularly use deep learning that will help growers and farmers control and monitor crops during the growing season; these tools help to extract meaningful information from large-scale aerial images received from the field using several techniques in order to create a strategic analytics for making a decision. The information result of the operation could be exploited for many reasons, such as sub-plot specific weed control. Our focus in this paper is on weed identification and control in sugar beet fields, particularly the creation and optimization of a Convolutional Neural Networks model and train it according to our data set to predict and identify the most popular weed strains in the region of Beni Mellal, Morocco. All that could help select herbicides that work on the identified weeds, we explore the way of transfer learning approach to design the networks, and the famous library Tensorflow for deep learning models, and Keras which is a high-level API built on Tensorflow.

Keywords

Deep learning, CNN, precision agriculture, decision making, strategic analytics.

Jabir, B., Falih, N. Sarih, A. and Tannouche, A. (2021) “A Strategic Analytics Using Convolutional Neural Networks for Weed Identification in Sugar Beet Fields", *AGRIS on-line Papers in Economics and Informatics*, Vol. 13, No. 1, pp. 49-57. ISSN 1804-1930. DOI 10.7160/aol.2021.130104.

Introduction

Today, companies can utilize machine learning or deep learning to help machines recognize the various aspects of agricultural production and help farmers enter the world of precision agriculture for making a decision (Fountas et al., 2006).

The sugar beet industry is one of the most important productive sectors in Beni Mellal region in Morocco (El Housni et al., 2020). A large area of 15,100 hectares was cultivated during the last agricultural season, which contributes around 26 percent of the national sugar production and plays an essential role in local economic development (El Bouzaidi et al., 2020). Successful cultivation of sugar beet depends mostly on weed control's efficacy during the first five to nine weeks after planting. It is known that the spread of weeds weakens the growth of sugar beets and hinders them from benefiting from the mineral salts necessary for better growth; therefore, weed control remains important for the increase in the productivity of the crops. Chemical control has become the most widely used means of controlling the spread of these weeds; this results in higher costs, pollution of soil and water resources. Besides, the herbicides may adversely affect the crops if applied in high concentrations, and then a frugal use of herbicides is desired (Jursik et al., 2020). So far, the common approach is the uniform application of herbicides to a field, neglecting the spatial variability of weed species and densities by mapping different species of weeds, their density and distribution; herbicide spraying can be adjusted as opposed to the uniform application.

This paper aims to use deep learning techniques to extract features from a set of collected images in order to classify in aerial images acquired; so we propose Convolutional Neural Networks (CNNs) for weeds detection (Jiang et al., 2020). We adapt and optimize our CNN model trained on our data sets and fine-tune with the five types of weeds most popular in sugar beet fields in the region; those weeds are stored in Kaggle public dataset (Hin, 2020). These approaches could be utilized as a part of a business-driven analytics strategy for agriculture organizations to develop visions and make a decision (Jabir and Falih, 2020).

CNN and Transfer learning approaches

Convolutional neural networks are the most powerful models for classifying images, also called CNN or ConvNet. They have two distinct parts. As input, a picture is provided in the form of an array of pixels. It has two dimensions for a grayscale image. The color is represented by a third dimension of depth, 3 to represent the fundamental colors [Red, Green, and Blue] (Bolo et al., 2019).

The first part of a CNN is the actual convolutional layer. It works as a feature extractor from images (Chen et al., 2016). An image is passed through a succession of filters, or convolution core, creating new images called convolution maps. The resolution of the image is reduced by a maximum local operation of some intermediate filters reduce. Finally, the convolution maps are flattened and concatenated into a vector of features called the CNN code (Albawi et al., 2017).

Convolutional neural networks are based on MLP (Multilayer Perceptron) and inspired by the behavior of the visual cortex of vertebrates. Although useful for image processing, MLPs find it very difficult to handle large images due to the exponential increase in the number of connections with the image size (Taud and Mas, 2018). In our example, we have an image with 98x98x3 size (98 width, 98 height, and 3 color channels), a single fully-connected neuron in the first hidden layer of the MLP would have 28812 entries (98 * 98 * 3) multiplied by the number of neurons would become enormous. A stack of independent processing layers forms a CNN architecture (Figure 1):

- CONV: The convolutional layer which processes the data of a receptive part.
- POOL: The pooling layer allows information to be compressed by reducing the intermediate image size (often by subsampling).
- RELU: The correction layer, frequently called by abuse 'ReLU', refers to the function activation (linear rectification unit).
- FC: The "fully connected" layer, which is a perceptron-type layer.
- LOSS: Loss layer.

Transfer learning is a process that facilitates learning from previous learning, provided there is compatibility between the two learnings (Zhuang et al., 2020). The typical use case of transfer learning is when the data are limited (a thousand photos of weeds) and where you learn from a neural network that has learned to recognize thousands of images (possibly involving different types of weed) classify weed species (Kartal et al., 2020).

Materials and methods

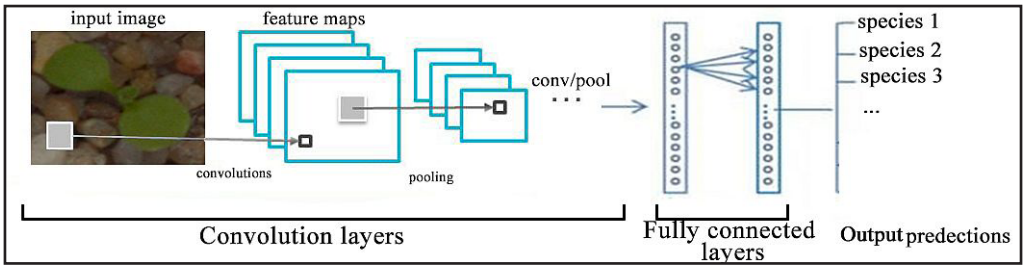
The data used for this study contain images of five different weed species well known in this region and spread in other parts of the world. In addition to the sugar beet pictures, to distinguish it from other herbs as shown below (Table 1).

Scientific name	English name	Moroccan name	Number of images
Beta vulgaris subsp	Sugar Beet	Chmander	463
Capsella	shepherd's purse	Kiss rai	274
Chenopodium	Fat Hen	Berramram	538
Galium aparine	Cleavers	Lassika	335
Sinapis arvensis	Charlock	kalkaz	452
Tripleurospermum	Scentless Mayweed	Kraa djaj	607

Source: authors

Table 1: A list of the weeds classes available for the study.

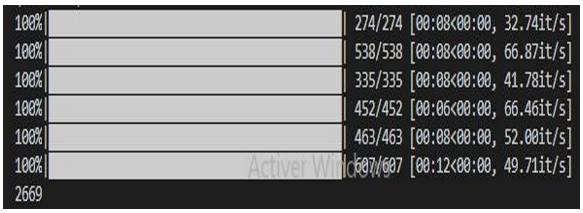
A digital camera (Sony a 6000) was used to manually collect our field images under different lighting conditions (from morning to afternoon in sunny and cloudy weather) from sugar beet fields of Fkihbensalh in Morocco. We combined these images with field images taken from an online repository Kaggle dataset (Analide



Source: authors

Figure 1: Standard architecture of a convolutional neuron network.

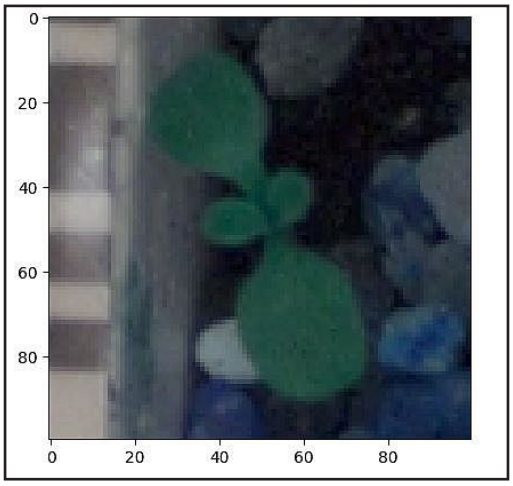
and Kim, 2017) to train the developed model, as shown in the following sections. We only use pictures of five harmful herbs that were strained in Beni Mellal area and added our images to them, in addition to pictures of sugar beet to get 2669 images in total belonging to six different classes (Figure 2), each class contains RGB images that show plants at different growth stages. The images are in various sizes and are in png format.



Source: authors

Figure 2: The six classes of the experiment.

Images in the training dataset had varying sizes. Consequently, images had to be resized before being used as input to the CNN model. Input images were resized to the shape of 98×98 pixels (Figure 3). Then created a path for each class and a variable that could store images in an array type (Figure 4).



Source: authors

Figure 3: Resized image.

```
[[ 86  84  86 ...  29  33  39]
 [109 105 106 ...  37  37  37]
 [135 135 134 ...  39  41  38]
 ...
 [158 157 155 ...  47  41  42]
 [158 157 157 ...  44  43  44]
 [158 157 157 ...  43  42  38]]
```

Source: authors

Table 1: A list of the weeds classes available for the study.

Implementation of CNN

This chapter defines the architecture of model that we created, and then we apply this model on images from our dataset. Thus, we work with python language for data science (Jarolímek et al., 2019), the Tensorflow and Keras libraries for learning and classification (Grattarola and Alippi, 2020). To improve the model's performance, we use some simple and effective techniques such as data augmentation and Tensorboard.

Tensorflow

Tensorflow is an Open Source programming framework intended for numerical computation made by Google in November 2015 (Goldsborough, 2016). Since its release, Tensorflow has continued to gain popularity and is becoming one of the most adopted frameworks for Deep Learning. Its name is notably inspired by the fact that the current operations on neural networks are mainly performed via multi-dimensional data tables, called Tensors. A two-dimensional Tensor is the equivalent of a matrix. Today, Google's main products are based on Tensorflow: Gmail, Google Photos, Voice Recognition.

Keras

Keras is a high-level neural networks API, written in Python, and can run on Theano or Tensorflow. It was developed with an emphasis on rapid experimentation, prepared to turn the idea into a result with the least amount of time. Keras was developed as part of the research effort of the ONEIROS (Open-ended Neuro Electronic Intelligent Robot Operating System) project, and its main author is a Google engineer called François Chollet (Chollet and Allaire, 2018).

In 2017, the Tensorflow organization at Google decided to support Keras in the Tensorflow core library. Chollet explained that Keras was created as an interface rather than an end-to-end learning environment. It presents a higher-level, more intuitive set of abstractions that make it easy to configure neural networks independently from the backend computer library. Microsoft is also working on joining a CNTK backend to Keras as well (Chollet, 2018).

Python

Python is an interpreted and object-oriented high-level programming language with dynamic semantics (no compilation step). It emerged as a popular language used a lot in data science

applications; take the case of the tech giant Google that has created a deep learning framework called Tensorflow (Nagpal and Gabrani, 2019).

This language is the first used for creating those frameworks' reusability of codes. Python and its libraries are available in source or in binaries free for most platforms and could be redistributed for free.

The architecture of our initial CNN model

As shown in Figures 5 and 6, during our experiments, we created a basic model. It is composed of two layers of convolution, two layers of Maxpooling, and two fully-connected layers (Abd El-Rahiem et al., 2019).

The input image is 98*98 size; the image goes to the first convolution layer (CONV). This layer is composed of 256 filters of size 3*3, the activation function ReLU (Rectified Linear Units) is used; This function ($F(x) = max(0,x)$) forces the neurons to return positive (Agarap, 2018). After this convolution, 256 feature maps of size 98*98 are created. Maxpooling (POOL) is applied afterward to diminish image size and settings. At the exit of this layer, we have 256 feature maps of size 49*49. We repeat the same thing with convolutional layer number two (composed of 256 filters). The ReLU activation function is always applied to each convolution. A Maxpooling layer is deployed after the convolution layer number two at the output, and we have 256 feature maps of size 23 * 23. The vector of characteristics resulting from the convolutions has a dimension of 135424.

Next, we use a neural network made up of two fully connected layers (FC) (Basha et al., 2020). The first layer has 64 neurons, and the second layer is a Softmax, which allows us to calculate the probability distribution of our classes (six classes in the dataset) (Figure 5 and 6).

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 98, 98, 256)	2560
activation (Activation)	(None, 98, 98, 256)	0
max_pooling2d (MaxPooling2D)	(None, 49, 49, 256)	0
conv2d_1 (Conv2D)	(None, 47, 47, 256)	590080
activation_1 (Activation)	(None, 47, 47, 256)	0
max_pooling2d_1 (MaxPooling2	(None, 23, 23, 256)	0
flatten (Flatten)	(None, 135424)	0
dense (Dense)	(None, 64)	8667200
dense_1 (Dense)	(None, 6)	390
activation_2 (Activation)	(None, 6)	0

Total params: 9,260,230
Trainable params: 9,260,230
Non-trainable params: 0

Source: authors

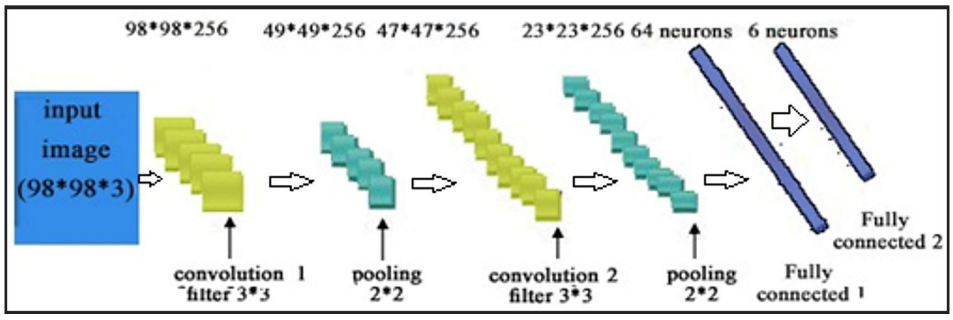
Figure 6: Model's configuration.

Results and discussion

There are several parameters employed for evaluating the performance of the classification and prediction models. Accuracy and error are some of the metrics for evaluating the performance of classification models (Nakzawa and Kulkarni, 2018). Accuracy is defined as follows:

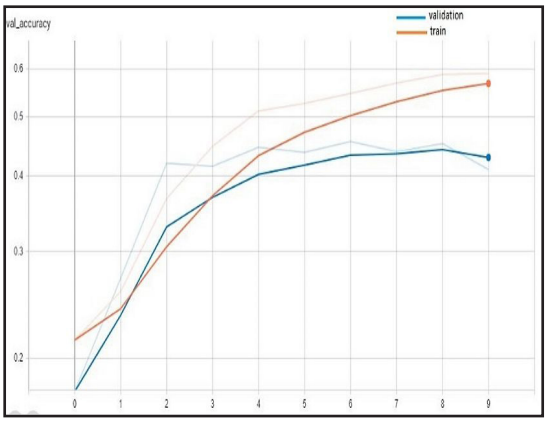
$$Accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ predictions}$$

Classification error relative number of misclassified examples, in other words, the percentage of wrong and incorrect predictions (Khaki et al., 2020). In order to show the results obtained for our model, we illustrate the results in terms of accuracy and error in the next section (Figures 7 and 8).



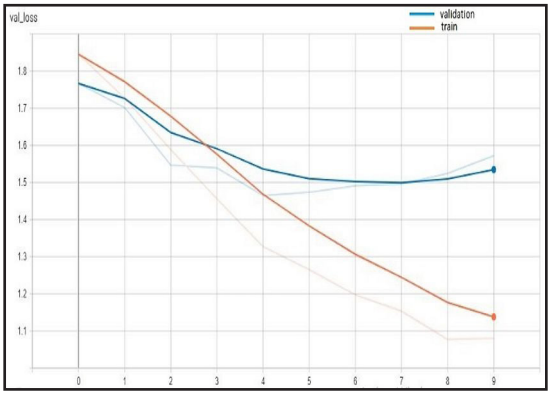
Source: authors

Figure 5: Model's architecture.



Source: authors

Figure 7: Model's precision.



Source: authors

Figure 8: Model's error.

After analyzing the results obtained, the following remarks are noted:

According to the previous two figures, the accuracy of the learning increased with the number of epochs. This reflects that in each epoch, our model learns more information to reach 73% accuracy. However, it eventually begins to fall after the 8th epoch. This should alert us that we are almost certainly starting to overfit (Xu et al., 2019). The reason why this happens is that the model is constantly trying to decrease in-sample loss. At some point, the model begins just to memorize input data instead of learning general things about the actual data, which means any new data we attempt to feed the model will perform poorly. Likewise, the learning and validation error decrease with the number of epochs, and after the 8th epoch start to rise. Thus, we will need more information to teach our model. Therefore we have to look for a solution that reduces overfitting, thus increasing the number of epochs and modifying other parameters, which is the object of the next sequence.

Optimizing the model

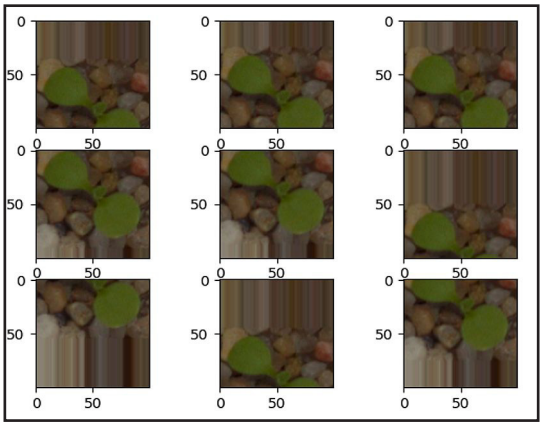
After some epochs, we have 73% validation accuracy, but we should probably discuss how we can optimize this model to improve this value. As we discussed in the previous sequence, we need to reduce overfitting, and for that, there are a lot of solutions. We quote five techniques (Data Augmentation, Regularization, Early Stopping, Simplifying the Model, Dropout) (Kim and Kang, 2020). For our case, we will focus on the data-augmentation solution. Also, we will use Tensorboard, which is a powerful application that comes with Tensorflow. It helps us visualize our models as they are trained and build a workflow to optimize our model's architecture.

Data augmentation

Data augmentation is a powerful technique that can reduce overfitting in Neural Networks, where we increase the number of training data using information only in our training data. This is done by applying domain-specific ways to examples from the training data that create a large amount of data (new and different training examples) (Hand et al., 2018). A variety of methods are supported, we focus on our model on five main types of data augmentation techniques for image data specifically:

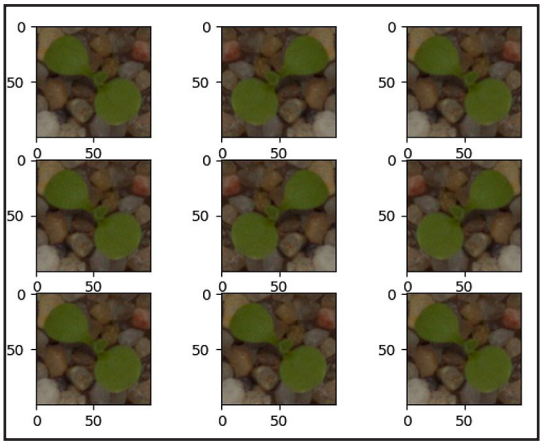
- Image zoom argument: *zoom_range*.
- Image shifts Arguments: *width _shift_range and height_shift_range.*
- Image rotations argument: *rotation_range*
- Image flips arguments: *horizontal_flip and vertial_flip*
- Image brightness argument: *brightness_ range.*

The following figures (Figures 9 and 10) are examples of two techniques used in our data augmentation. The first is vertical shifts of the image via the *height_shift_range* argument; in this case, we specified the percentage of the image to shift to 0.5 the image's height. The second example is an augmenting of the chosen image with horizontal flips via the *horizontal_flip* argument.



Source: authors

Figure 9: Images with a random vertical shift.



Source: authors

Figure 10: Images with a random horizontal flip.

Tensorboard

Convolutional neural networks, however, have several parameters to adjust. The most basic things for us to modify are layers and nodes per layer and 0, 1, or 2, dense layers, and other parameters like varying layer sizes, learning rates, and activation functions. Thanks to the python code, we create a loop that allows us to assemble and test a set of combinations (80 combinations) (Figure 11) and choose the best of them, which gives good results (high accuracy visualized on Tensorboard application (Luus et al., 2019) (Figure 12).

```
dense_layers = [0,1,2,3]
layer_sizes = [32,64,128,256,512]
conv_layers = [1,2,3,4]

for dense_layer in dense_layers:
    for layer_size in layer_sizes:
        for conv_layer in conv_layers:
            NAME = "{}-conv-{}-nodes-{}-dense-{}".format(conv_layer, layer_size, dense_layer, int(time.time()))
            print(NAME)
            model = Sequential()
            model.add(Conv2D(layer_size, (3, 3), input_shape=X.shape[1:]))
            model.add(Activation('relu'))
            model.add(MaxPooling2D(pool_size=(2, 2)))
            for l in range(conv_layer-1):
                model.add(Conv2D(layer_size, (3, 3)))
            ...

            for _ in range(dense_layer):
                model.add(Dense(layer_size))
                model.add(Activation('relu'))
            ...
```

Source: authors

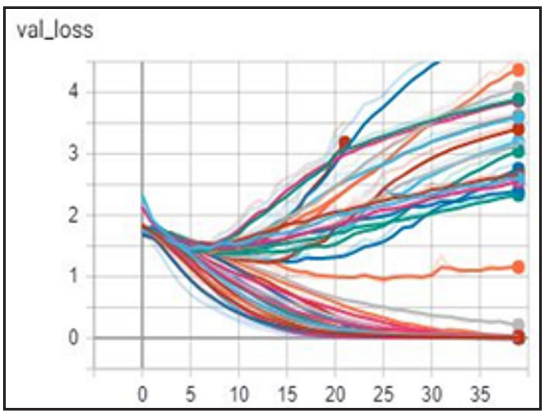
Figure 11: A part of python code, which forms the combinations.

```
1-conv-32-nodes-0-dense-1599243547
2-conv-32-nodes-0-dense-1599243547
3-conv-32-nodes-0-dense-1599243547
4-conv-32-nodes-0-dense-1599243547
1-conv-64-nodes-0-dense-1599243547
2-conv-64-nodes-0-dense-1599243547
...
3-conv-256-nodes-3-dense-1599243547
4-conv-256-nodes-3-dense-1599243547
1-conv-512-nodes-3-dense-1599243547
2-conv-512-nodes-3-dense-1599243547
3-conv-512-nodes-3-dense-1599243547
4-conv-512-nodes-3-dense-1599243547
```

Source: authors

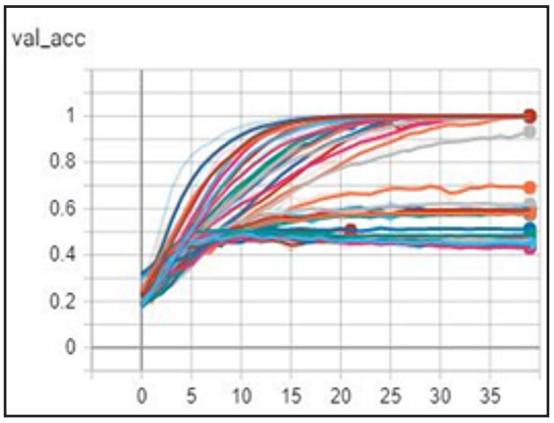
Figure 12: Set of 80 created and tested combinations .

The figures (Figures 13 and 14) below captured from Tensorboard show a visualization of the 80 models created above. We could be tempted to take the highest validation accuracy model, but instead, we tend to go for the lowest validation loss models in the resulting graph. Therefore, zooming into the validation accuracy graph or validation loss graph allowed us to choose and check some of the best models.



Source: authors

Figure 13: Error of the generated models.



Source: authors

Figure 14: Accuracy of the generated models.

After zooming into the graph (Figure 15), here are the top 3 models have the highest accuracy. We can be satisfied with 0 dense, and 1 convolutional layer, and 128 nodes per layer.

Source: authors

Figure 15: Top three models with the highest accuracy.

Changing some parameters, using a model with data augmentation technique and choose 0 dense, 1 convolutional layer and 128 nodes per layer, with more epochs (40), had developed the precision to 82% of accuracy with 20 hours of training (Figure 16).

```
----------------
84/84 [==============================]- 11s 127ms/step -loss: 0.5166 - sparse_categorical_accuracy: 0.8277
----------------
Test accuracy: 0.8276507854461670
----------------
```

Source: authors

Figure 16: Test accuracy of the optimized model.

Also, increasing the number of pictures can increase the model's accuracy, knowing that there is no specific number of model training images, but previous experiences affirm that when we have more images, we will have more accuracy. According to the results discussed above, the experiments have demonstrated good achievement in optimizing and training our CNN model to address weed classification with limited training images. But there are always limitations which can be cited as follows:

- The resolution of training images poses some restrictions due to memory requirements.
- The models trained in images where the weeds are in determining size have difficulties in detecting the weeds when the size is significantly different.
- The optimization with Tensorboard consumes much time and resources.
- Developing systems that can take pictures that will be used to determine the percentage of weeds and their spread still requires deep scientific research.

Conclusion

Our objective was to create and optimize a CNN model able to detect and identify the percentage and the species of weed most popular in this area, with high accuracy. That could promote the frugal use of herbicides and control the spread of these weeds. In the proposed work, we created a CNN model that can classify images from a large dataset. This model gives an accuracy of 82% after a set of processes that help improve precision and decrease error. Indeed, the model is primarily focused on species common to the studied region (weeds that are in the dataset), but since it has reached a good accuracy, we can feed the dataset with other vegetation, and obviously, it can give good results.

Further steps may be taken to improve the accuracy and evaluate the model with other parameters in the future. Also, we need to try to find a solution to the real-time image capturing and combine it with our CNN model as a powerful real-time weed-crop classification and technical localization for robotic weed control.

Corresponding authors
Brahim Jabir
Faculty of Sciences and Technics, Sultan Moulay Slimane University, Beni Mellal 23000, Morocco
Phone: +212 639 08 05 91, E-mail: ibra.jabir@gmail.com

References

[1] Abd El-Rahiem, B., Ahmed, M. A. O., Reyad, O., Abd El-Rahaman, H., Amin, M. and Abd El-Samie, F. (2019) "An efficient deep convolutional neural network for visual image classification", In *International Conference on Advanced Machine Learning Technologies and Applications,* Springer, Cham, pp. 23-31. DOI 10.1007/978-3-030-14118-9_3.

[2] Agarap, A. F. (2018) "Deep learning using rectified linear units (relu)", arXiv preprint arXiv:1803.08375.

[3] Albawi, S., Mohammed, T. A. and Al-Zawi, S. (2017) "Understanding of a convolutional neural network", In 2017 *International Conference on Engineering and Technology (ICET), Antalya, Turkey,* IEEE, pp. 1-6. DOI 10.1109/ICEngTechnol.2017.8308186.

[4] Analide, C. and Kim, P. (2017) "Experiential learning in data science: from the dataset repository to the platform of experiences", In *Intelligent Environments 2017: Workshop Proceedings of the 13th International Conference on Intelligent Environments,* IOS Press, Vol. 22, 122 p.

[5] Basha, S. S., Dubey, S. R., Pulabaigari, V. and Mukherjee, S. (2020) "Impact of fully connected layers on performance of convolutional neural networks for image classification", *Neurocomputing*, Vol. 378, pp. 112-119. ISSN 0925-2312. DOI 10.1016/j.neucom.2019.10.008.

[6] Bolo, B., Mpoeleng, D. and Zlotnikova, I. (2019) "Development of Methods Acquiring Real Time Very High Resolution Agricultural Spatial Information Using Unmanned Aerial Vehicle", *AGRIS on-line Papers in Economics and Informatics,* Vol. 11, No. 2, pp. 21-29. ISSN 1804-1930. DOI 10.7160/aol.2019.110203.

[7] El Bouzaidi, H., Hafiane, F. Z. and Fekhaoui, M. (2020) "Inventory of Pesticides and their impact on the environment by calculating the frequency of treatment indicator in the Gharb plain (Morocco)", *Mediterranean Journal of Chemistry*, Vol. 4, No. 10. ISSN 2028-3997.

[8] El Housni, Z., Ezrari, S., Tahiri, A. and Ouijja, A. (2020) "Resistance of Cercospora beticola Sacc isolates to thiophanate methyl (benzimidazole), demethylation inhibitors and quinone outside inhibitors in Morocco", *EPPO Bulletin*, Vol. 50, No. 2, pp. 350-357. E-ISSN 1365-2338. DOI 10.1111/epp.12673.

[9] Fountas, S., Wulfsohn, D., Blackmore, B. S., Jacobsen, H. L. and Pedersen, S. M. (2006) "A model of decision-making and information flows for information-intensive agriculture", *Agricultural Systems*, Vol. 87, No. 2, pp. 192-210. ISSN 0308-521X. DOI 10.1016/j.agsy.2004.12.003.

[10] Goldsborough, P. (2016) "A tour of tensorflow", arXiv preprint arXiv:1610.01178.

[11] Grattarola, D. and Alippi, C. (2020) "Graph neural networks in tensorflow and keras with spectral", arXiv preprint arXiv:2006.12138.

[12] Han, D., Liu, Q. and Fan, W. (2018) "A new image classification method using CNN transfer learning and web data augmentation", *Expert Systems with Applications,* Vol. 95, pp. 43-56. ISSN 0957-4174. DOI 10.1016/j.eswa.2017.11.028.

[13] Hin, D. (2020) "StackOverflow vs Kaggle: A Study of Developer Discussions About Data Science", arXiv preprint arXiv:2006.08334.

[14] Chen, Y., Jiang, H., Li, C., Jia, X. and Ghamisi, P. (2016) "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 54, No. 10, pp. 6232-6251. ISSN 0196-2892. DOI 10.1109/TGRS.2016.2584107.

[15] Chollet, F. (2018) "Keras: The python deep learning library", ascl, ascl-1806.

[16] Chollet, F. and Allaire, J. J. (2018) "*Deep Learning mit R und Keras: Das Praxis-Handbuch von den Entwicklern von Keras und RStudio*", MITP-Verlags GmbH & Co. KG. ISBN 10 3958458939, ISBN 13 978-3958458932.

[17] Jabir, B. and Falih, N. (2020) "Digital agriculture in Morocco, opportunities and challenges", In *2020 IEEE 6th International Conference on Optimization and Applications (ICOA), IEEE,* pp. 1-5. IEEE. DOI 10.1109/ICOA49421.2020.9094450.

[18] Jarolímek, J., Pavlík, J., Kholova, J. and Ronanki, S. (2019) "Data Pre-processing for Agricultural Simulations", *Agris on-line Papers in Economics and Informatics*, Vol. 11, No. 1, pp. 49-53. ISSN 1804-1930. DOI 10.7160/aol.2019.110105.

[19] Jiang, H., Zhang, C., Qiao, Y., Zhang, Z., Zhang, W. and Song, C. (2020) "CNN feature based graph convolutional network for weed and crop recognition in smart farming", *Computers and Electronics in Agriculture*, Vol. 174, ISSN 0168-1699. DOI 10.1016/j.compag.2020.105450.

[20] Jursík, M., Soukup, J. and Kolářová, M. (2020) "Sugar beet varieties tolerant to ALS-inhibiting herbicides: A novel tool in weed management", *Crop Protection*, Vol. 137, ISSN 0261-2194. DOI 10.1016/j.cropro.2020.105294.

[21] Kartal, S., Choudhary, S., Stočes, M., Šimek, P., Vokoun, T. and Novák, V. (2020) "Segmentation of Bean-Plants Using Clustering Algorithms", *AGRIS On-line Papers in Economics and Informatics,* Vol. 12, No. 3, pp. 36-43. ISSN 1804-1930. DOI 10.7160/aol.2020.120304.

[22] Khaki, S., Wang, L. and Archontoulis, S. V. (2020) "A CNN-RNN Framework for Crop Yield Prediction", *Frontiers in Plant Science*, Vol. 10. E-ISSN 1664-462X. DOI 10.3389/fpls.2019.01750.

[23] Kim, H. C. and Kang, M. J. (2020) "A comparison of methods to reduce overfitting in neural networks", *International Journal of Advanced Smart Convergence*, Vol. 9, No. 2, pp. 173-178. E-ISSN 2288-2855. ISSN 2288-2847. DOI 10.7236/IJASC.2020.9.2.173.

[24] Luus, F., Khan, N. and Akhalwaya, I. (2019) "Active Learning with TensorBoard Projector", arXiv preprint arXiv:1901.00675.

[25] Nagpal, A. and Gabrani, G. (2019) "Python for Data Analytics, Scientific and Technical Applications," *Amity International Conference on Artificial Intelligence (AICAI)*, Dubai, United Arab Emirates, 2019, pp. 140-145. DOI 10.1109/AICAI.2019.8701341.

[26] Nakazawa, T. and Kulkarni, D. V. (2018) "Wafer map defect pattern classification and image retrieval using convolutional neural network", *IEEE Transactions on Semiconductor Manufacturing*, Vol. 31, No. 2, pp. 309-314. E-ISSN 1558-2345, ISSN 0894-6507. DOI 10.1109/TSM.2018.2795466.

[27] Taud, H. and Mas, J. (2018) "Multilayer Perceptron (MLP)", In: Camacho Olmedo M., Paegelow M., Mas JF., Escobar F. (eds) *Geomatic Approaches for Modeling Land Change Scenarios. Lecture Notes in Geoinformation and Cartography.* Springer, Cham. DOI 10.1007/978-3-319-60801-3_27.

[28] Xu, Q., Zhang, M., Gu, Z. and Pan, G. (2019) "Overfitting remedy by sparsifying regularization on fully-connected layers of CNNs", *Neurocomputing*, Vol. 328, pp. 69-74. ISSN 0925-2312. DOI 10.1016/j.neucom.2018.03.080.

[29] Zhang, S., Zhang, X., Chan, J. and Rosso, P. (2019) "Irony detection via sentiment-based transfer learning", *Information Processing & Management*, Vol. 56, No. 5, pp. 1633-1644. ISSN 0306-4573. DOI 10.1016/j.ipm.2019.04.006.