DETECTION OF HARMFUL INSECTS FOR ORCHARD USING CONVOLUTIONAL NEURAL NETWORKS

Raluca TRUFELEA[1], Mihai DIMOIU[2], Loretta ICHIM[3], Dan POPESCU[4]

Monitoring the Pentatomidae family of pests in in the modern agriculture allows researchers to spot differences in infection levels and improve the development of integrated pest management strategies. The capacity of deep learning models to classify pest species with increased interspecies similarity and intraspecies variability was explored in this paper. For detection of four species of Phyllocephalidae insects, a modified SSD model having the IoU value of 70.2% as performance indicator was proposed in this paper. As a consequence, this procedure might save costs, improve performances, and make the analysis more scalable.

Keywords: convolutional neural networks, single shot detector, object detection, image segmentation, harmful insects

1. Introduction

The widespread of harmful insects in the world is favored by the movement of goods and people (for example, the appearance of *Halyomorpha Halys* in Europe). These insects cause great damage to agricultural producers and, as a result, there are intense concerns to monitor and stop their spread by ecological methods. Today, computer vision technology is frequently used to identify insects and the diseases they cause as well as to monitor their spread in crops or orchards [1]. For insect detection on large areas, a variety of new remote sensing technologies such as unmanned aerial vehicles (UAV), high resolution RGB cameras, thermal imaging sensors, and multispectral cameras, are available on the market. In modern approaches to image processing and analysis for the recognition of plant diseases and invasive insects, machine learning techniques and, especially, the artificial neural networks are increasingly used. To provide the best model accuracy, one of the most database used is the Maryland Biodiversity

[1] Student, Faculty of Automatic Control and Computers, University POLITEHNICA of Bucharest, Romania, e-mail: trufelearaluca@yahoo.com
[2] PhD student, Faculty of Automatic Control and Computers, University POLITEHNICA of Bucharest, Romania, e-mail: mihaidimoiu@yahoo.com
[3] Prof., Faculty of Automatic Control and Computers, University POLITEHNICA of Bucharest, Romania, e-mail: loretta.ichim@upb.ro
[4] Prof., Faculty of Automatic Control and Computers, University POLITEHNICA of Bucharest, Romania, e-mail: dan.popescu@upb.ro

and Global Biodiversity Information Facility which offers *Halyomorpha Halys* [2] and related species images. The Pentatomidae family of insects is part of the Hemiptera family and consists approximately 900 genera and 4700 species. Cotton, sorghum, soybeans, native and ornamental trees, shrubs, vines, wildflowers, and many cultivated crops are all threatened by this species [3]. The population of this invasive species has grown significantly, though, some species are considered beneficial. The authors of this paper proposed that a CNN be trained to recognize four different kinds of Pentatomidae insects, including *Halyomorpha Halys*. Since the damages with *Halyomorpha Halys* are becoming more and more significant, this paper demonstrates the possibilities of employing a computer vision system for recognition of pests from images. Thus, the objective of this paper is to identify and categorize insects belonging to the Pentatomidae family for economic purposes [4].

PestNet [5] has been developed as a strategy for pest management. It has one of the highest detection accuracies of all the insects. The researchers used a wide variety of data with 80k images and 580k insects divided into 16 classes, one for each class. The proposed method achieved an average accuracy of 75.45%. PestNet is a network architecture that supplies crop information, including plant pest control.

Object detection using neural networks is considered one of the advanced stages in the computer vision. Its purpose is to obtain accurate location of the object in the image. Notable architectures are Faster R-CNN [6], YOLO [7] and SSD [8]. Feature extraction and pattern recognition are the two most critical stages. Feature extraction is the process of extracting information from images as feature vectors or feature maps. Pattern recognition is used to train the model to classify input images across categories. CNN known as Convolutional Neural Networks, in deep learning has surpassed computer vision in the generic [9] object detection, as is widely known.

By combining a sparse-coding strategy for encoding insect pictures with a multiple-kernel learning (MKL) technique, authors in [10] designed an insect identification system that achieved a mAP (mean average precision) of 85.5 % on 24 common insects in crop fields. However, the approach in [10] requires multi-image preprocessing, such as image denoising and segmentation, which involves a substantial amount of time and technical expertise, thus predictions based on images without preparation may be insufficient. In [11] an insect classification system is developed, which has been modified to improve performance.

In this paper, we demonstrate the importance of being able to locate insects in images using Convolutional Neural Networks (CNN) and, especially, a modified Single Shot Detector (SSD). The goal of this study is to identify four different types of insects to decrease the damage they do and enhance production.

2. Materials and methods

For detection of four species of Phyllocephalidae insects, a modified SSD model having avalue of 70.2% IoU was created (Fig. 1). The RGB images used have a resolution of 300 × 300 pixels. For image processing, several filters were applied to the original images, including random saturation, hue, contrast, scale, rotation, and flip. The spacing interval for filters was carrefuly chosen. In addition, Max Pooling layers are used. Max Pooling extracts the maximum value of pixels in a specified square. We get good results by combining all of these operations in the proposed Neural Network and feeding the final output into a SoftMax function, which uses a gradient optimizer to train the model weights. During the learning phase, we also introduced random dropouts to the CNN to establish decision boundaries and classify images into one of the four classes established. As the data stream goes through the network, some parameters change, such as the filters and outputs, which become smaller as the process progresses.

This is due to the network's need to learn comparable characteristics at different scales. Although applying convolution filters to the whole training image implies that the CNNs are invariant of rotation and translation features, this makes the CNNs more forgiving of feature distortions in the pooling layers. For the application, we selected four species of insects, grouped in four classes, from the Phyllocephalidae family: Hayomorpha Halys adult, Halyomorpa Halys Nympha specie, Pyrrhocoris apterus, and Nezara viridula. A total of 760 images were used for training and validation: 600 for training and 160 for validation. Of these, 520 images are from the Maryland Biodiversity database and 240 from our own dataset. The segmentation, annotation, and labeling methods reflect the selection of the region of interest, in this case the insects. The dataset was placed in such a manner that each class had about the same amount of representatives.

As metioned previously, we modified a Single Shot Detector for insect classification. SSD needs only one shot to detect multiple objects within the image, while regional proposal networks like R-CNN uses two shots, one for generating regional proposals and one for detecting the object for each proposal. SSD is constructed from two parts, the backbone and detection.

The feature map at the top of the CNN corresponds to the lines in the original image, whilst those towards the end correspond to more descriptive features. Each convolutional layer in a CNN generates a feature map with the same size. Each feature map item corresponds to a specific location on the input image. The feature map becomes much more descriptive as the depth of CNN grows. The initial few layers of a CNN produce feature map pieces that correspond to small regions in the original image that are edges, lines, or corners.

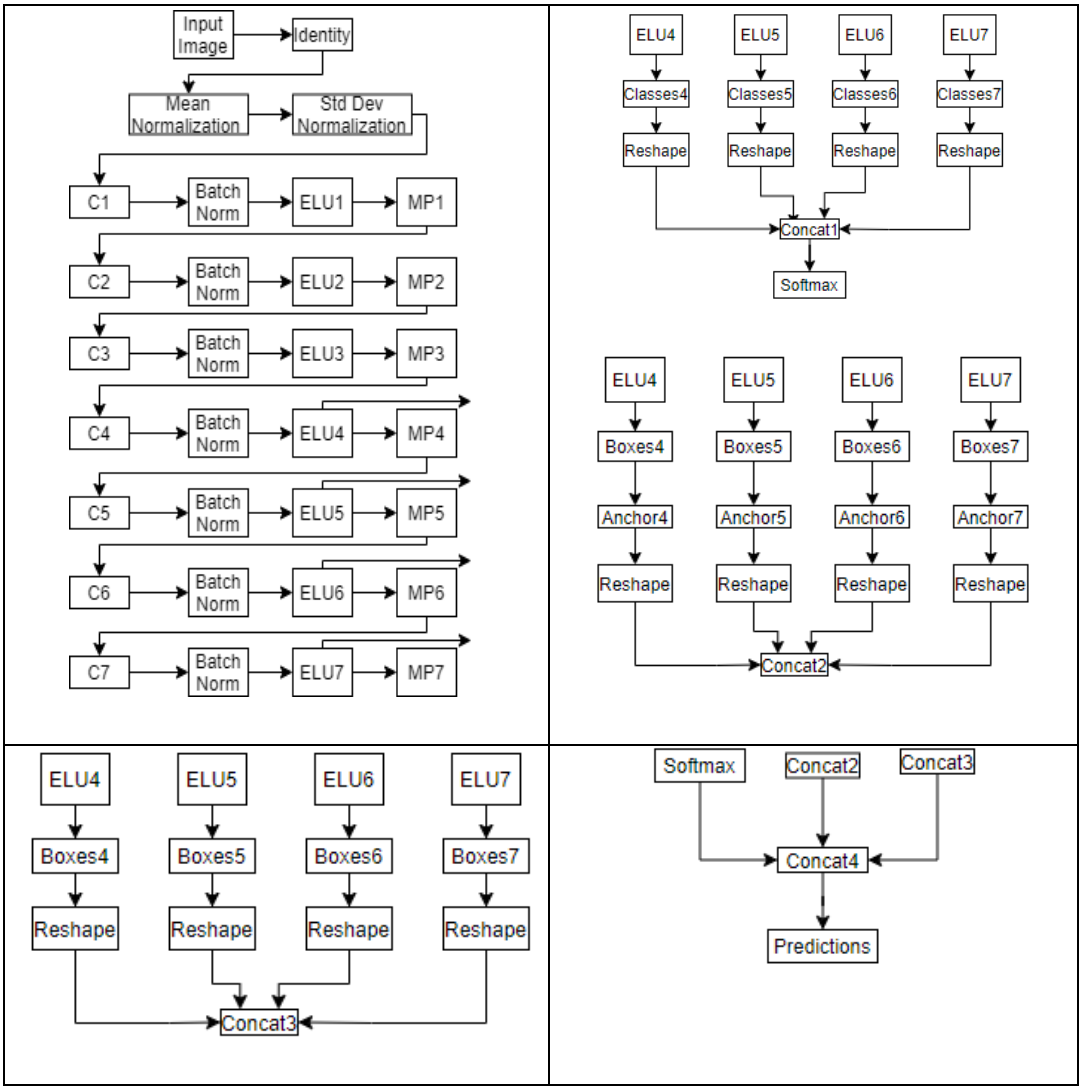


Fig. 1. SSD model used: C – convolution block, MP – MaxPooling, BN – batch normalization, R – Reshape, Con – Concatenation, A – Activation, S – SoftMax, and AB – Anchor Boxes

Further down the CNN, the feature map pieces correspond to bigger regions on the input image. Those big patches might be parts of an object or whole items. A base network (also known as a backbone network) is a CNN network that extracts feature maps in Object Detection. They're mostly CNNs that are used to solve image classification challenges. The backbone is responsible for extracting feature maps from the images that we want to detect objects. The original SSD's backbone is modified version of VGG-16 neural network extended by a few convolutional feature layers. This implementation has 7 convolutional

layers and 4 convolutional predictor layers that take their input from layers 4, 5, 6, and 7.

Table 1

Backbone network description

Layer Input	Size Output	Kernel Size	Filters
Image	300x300x3		
Identity	300x300x3		
Mean Normalization	300x300x3		
Standard Deviation Norm	300x300x3		
Conv2D_1	300x300x32	5x5	32
BN_1	300x300x32		32
ELU_1	300x300x32		32
Max Pooling 2D_1	150x150x32		32
Conv2D_2	150x150x48	3x3	48
BN_2	150x150x48		48
ELU_2	150x150x48		48
Max Pooling 2D_2	75x75x48		48
Conv2D_3	75x75x64	3x3	64
BN_3	75x75x64		64
ELU_3	75x75x64		64
Max Pooling 2D_3	37x37x64		64
Conv2D_4	37x37x64	3x3	64
BN_4	37x37x64		64
ELU_4	37x37x64		64
Max Pooling 2D_4	18x18x64		64
Conv2D_5	18x18x48	3x3	48
BN_5	18x18x48		48
ELU_5	18x18x48		48
Max Pooling 2D_5	9x9x48		48
Conv2D_6	9x9x48	3x3	48
BN_6	9x9x48		48
ELU_6	9x9x48		48
Max Pooling 2D_6	4x4x48		48
Conv2D_7	4x4x48	3x3	48
BNn_7	4x4x48		48
ELU_7	4x4x48		48

After using the inputs from those 4 layers, we continued to stack two more predictor layers on the top of each of those layers: one for class prediction and one for box localization. The final prediction consists into batch_size, class, xmin, ymix, xmax, ymax. After each of 7 convolutional layers in the backbone network we used 2 additional layers: Batch Normalization and Exponential Linear Unit (ELU). We chose ELU instead of ReLU because it does not have the dying problem of ReLU and function tends to converge to zero the cost faster and produces more accurate results. One of the best practices for training a neural network is to normalize the input data to obtain a mean close to 0. By normalizing

the data, the model generally speeds up the learning phase and leads to faster convergence. By this method we avoid the numerical problem with very large and very small numbers. Also, by normalize the standard deviation helps the gradient descent solver. The Hessian matrix becomes much more stable and easier to traverse if all the inputs are scaled.

Table 2

Predictor network description

Layer	Layer Input	Size Output	Filters	Kernel Size
Classes_4	ELU_4	37x37x20	20	3x3
Classes_5	ELU_5	18x18x20	20	3x3
Classes_6	ELU_6	9x9x20	20	3x3
Classes_7	ELU_7	4x4x20	20	3x3
Boxes_4	ELU_4	37x37x16	16	3x3
Boxes_5	ELU_5	18x18x16	16	3x3
Boxes_6	ELU_6	9x9x16	16	3x3
Boxes_7	ELU_7	4x4x16	16	3x3
Classes_Reshape_4	Classes_4	5476x5		
Classes_Reshape_5	Classes_5	1296x5		
Classes_Reshape_6	Classes_6	324x5		
Classes_Reshape_7	Classes_7	64x5		
Anchors_4	Boxes_4	37x37x4x8		
Anchors_5	Boxes_5	18x18x4x8		
Anchors_6	Boxes_6	9x9x4x8		
Anchors_7	Boxes_7	4x4x4x8		
Classes_concat	Classes_Reshape_4, 5, 6, 7	7160x5		
Boxes_Reshape_4	Boxes_4	5476x4		
Boxes_Reshape_5	Boxes_5	1296x4		
Boxes_Reshape_6	Boxes_6	324x4		
Boxes_Reshape_7	Boxes_7	64x4		
Anchors_Reshape_4	Anchors_4	5476x8		
Anchors_Reshape_5	Anchors_5	1296x8		
Anchors_Reshape_6	Anchors_6	324x8		
Anchors_Reshape_7	Anchors_7	64x8		
Classes_Softmax	Classes_concat	7160x4		
Boxes_Concat	Boxes_Reshape_4, 5, 6, 7	7160x5		
Anchors_Concat	Anchors_Reshape_4, 5, 6, 7	7160x8		
Predictions	Boxes_Concat Classes_Softmax Anchors_Concat	7160x17		

By collecting the output from layers 4 to 7, we applied two predictors to each layer. Those predictors are convolutional layers. From box localization we attached anchor boxes. For class prediction we have used a SoftMax activation

function. We have reshaped those 3 tracks into a one hot vector encoding (class predictions, boxes localization, and anchors) and concatenated them into a single big layer for our final output. One component of a typical object detection pipeline is for producing classification proposals. Candidate regions for the object of interest are referred to as proposals.

The predictions are subsequently filtered out using a variety of filtering techniques. Authors of [12] used a four-step technique to decode the predictions generated by the SSD network: Bounding Boxes Decoding, Confidence Thresholding, Non-Max Suppression, and Top-K Filtering. SSD predictions are centroid encoded with a standard deviation. As a result, the first step is to decode the encoded predictions and convert them back to the *cx* and *cy* (width and height format). We need to eliminate bounding boxes with confidence scores lower than a specific threshold after decoding the bounding box predictions. This filtering procedure is carried out for each class.

Since the SSD network produces the class predictions through the SoftMax function, we can obtain the confidence score of a particular class by its position in the SoftMax output. This confidence score tells us how sure the model is that an object of that exists inside the bounding box. Because the SSD network generates class predictions using the SoftMax function, we may calculate the confidence score of a certain class based on its position in the SoftMax output. This confidence score indicates how certain the model is that an object of that kind exists within the bounding box. We need to combine overlapping boxes together after we have filtered out the bounding boxes whose class has a low confidence score. This is known as Non-Max Suppression (NMS). It contributes to further reduce the number of predictions by combining overlapping forecasts into a single prediction. Even after applying Confidence Score Thresholding and NMS to each class, the number of residual predictions might be enormous. However, because the number of items (of our interest) that might appear in a picture is restricted, the bulk of those predictions are unnecessary. As a result, we may rank those forecasts according to their confidence level and choose the *k* greatest confidence score. Top *K* selection yields *k* predictions. Each of the *k* forecasts has a different level of confidence. To create the final, we further narrow down the *k* forecasts by selecting only those with confidence scores over a specific threshold. Normally, this threshold is determined by selecting the one with the highest mAP during the model's assessment. We may combine them into one Keras layer. The benefit of establishing a Keras layer for the decoding process is that we can generate a model file with the decoding process built in. The localization loss between the predicted box l and the ground-truth box g is outlined as a smooth loss with corrections to the default bounding box. We used SSD’s loss function to assess the behavior of the network. The loss that the model produces for each sample or batch of samples is used to measure its performance

during training. Original SSD loss function [8] is presented in Eq. (1). The equation combines regression loss (L_{loc}) and classification loss (L_{conf}) with a scale factor α for localization, where N represents number of positive matches, c is the class, x is a coefficient equal of 1 only if IoU score is over 0.5, l represents the predicted box, and g is ground truth box.

$$\boldsymbol{L}(\boldsymbol{x},\boldsymbol{c},\boldsymbol{l},\boldsymbol{g}) = \frac{\mathbf{1}}{\boldsymbol{N}}(\boldsymbol{L_{conf}}(\boldsymbol{x},\boldsymbol{c})) + \boldsymbol{\alpha}\,\boldsymbol{L_{loc}}(\boldsymbol{x},\boldsymbol{l},\boldsymbol{g})) \quad (1)$$

The model has been trained using a local machine powered by a GPU NVIDIA GTX 1070. Model was trained with batches of 1 and 8. While training, we have defined callbacks for our model: checkpoint for saving only best weights, early stopping, and reduce learning rate. Model was trained for 20 epochs with training steps per epoch of train dataset divided by batch size. Furthermore, we have implemented a visualization technique to view each layer's output after each step.

3. Experimental results

Table III presents the species that we used in this paper, also the number of images per species used in training phase and the class name for each species. In Fig. 2 are presented images for training the model: a) *Halyomorpha Halys aduls*, b) *Pyrrhocoris Apterus*, c) *Halyomorpha Halys nympha*, and d) *Nezara Virdula*. The SSD model obtained an IoU (intersection over union) value of 70.2% and a value of 89% for ACC (accuracy), considering the average of the mentioned four classes. In the Fig. 3 there are some examples in the testing phase from the same species (a, b, c, and d) for batch size of 1 (a-1, b-1, c-1, d-1) and for batch size of 8 (a-8, b-8, c-8, d-8). The insects are framed with a rectangle with the color associated with the class and the probability of decision.

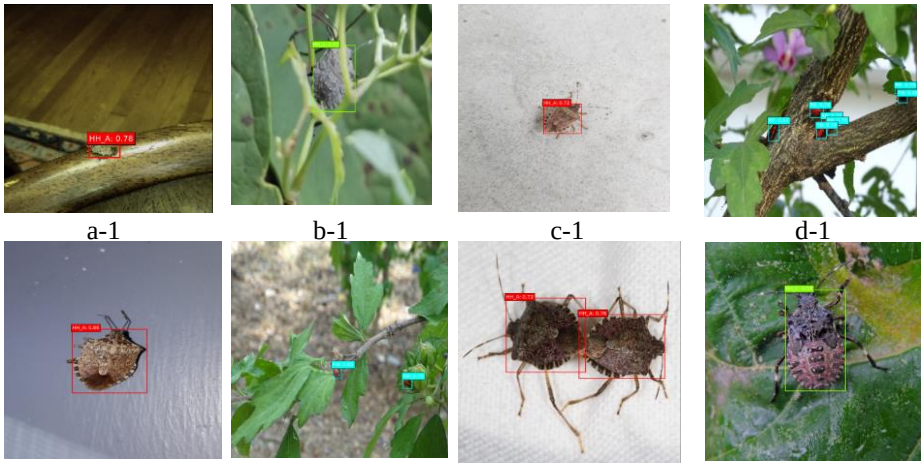
Table 3

Species used in training phase

Species	No. Images	Class Name
Halyomorpha Halys adult	400	HH_A
Halyomorpha Halys nympha	150	HH_C
Pyrrhocoris apterus	95	RB
Nezara viridula	115	BB

a) b) c) d)

Fig. 1. Examples of images from the training set – from each considered class.



a-1 b-1 c-1 d-1

a-8 b-8 c-8 d-8

Fig. 2. Experimental results for batch size of 1 (a-1, b-1, c-1, d-1) and for batch size of 8 (a-8, b-8, c-8, d-8).

As we mentioned above, we have implemented a method to visualize the output from the layers. In the Fig. 4, is described each filter from some backbone layers. As it can be noticed, going deeper into the neural network architecture, the convolutional layers learn to extract meaningful features from each layer. From the first and second layers it can be seen the network learned the position of the insect and the texture. We start with feature map with size of 300×300 (image size) and 32 filters. In last convolutional layer, we have 48 filters of feature map of 9×9. The input image for the SSD network in Fig. 4 is those in Fig. 2 a.

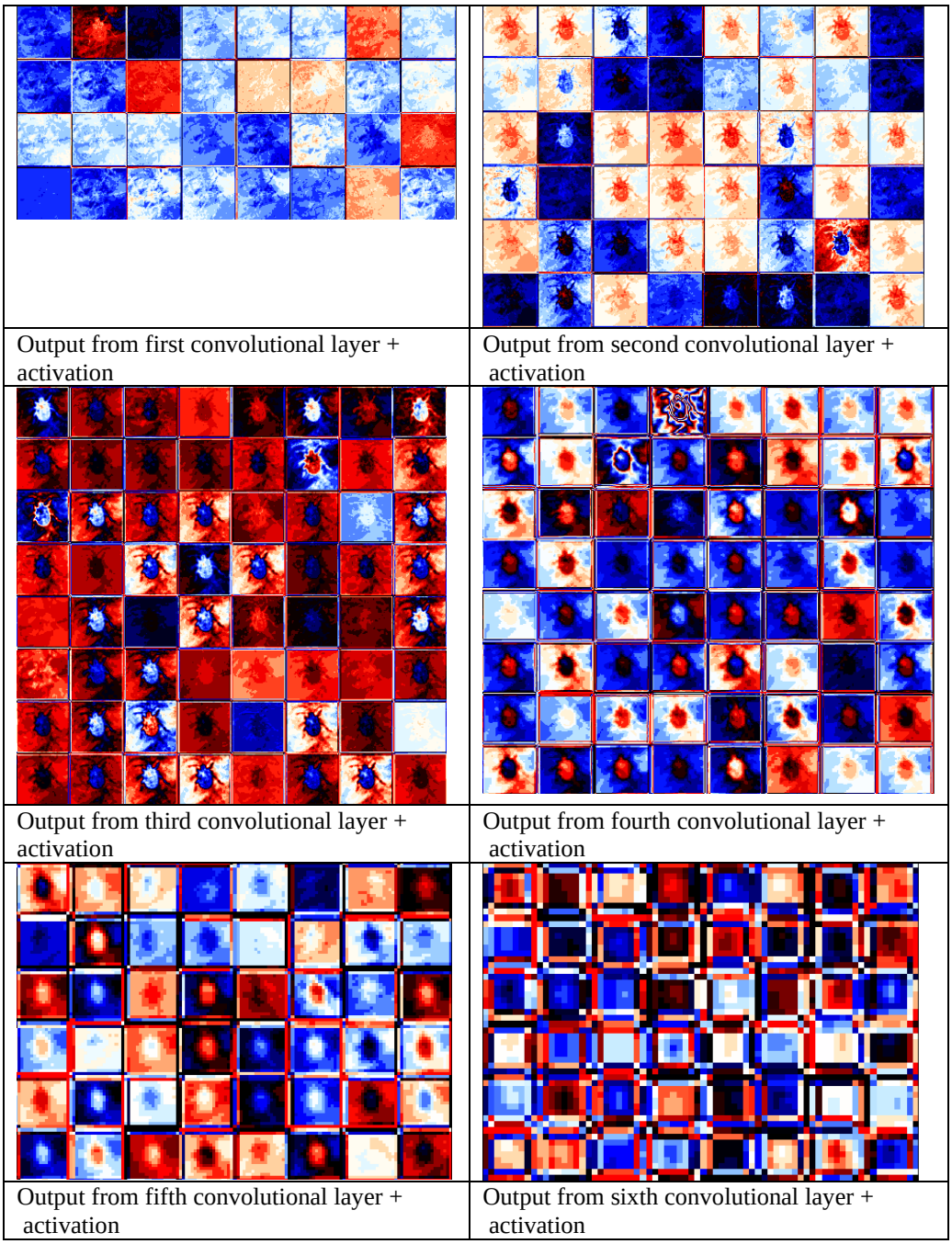


Fig. 4. Output from three layers to visualize what neural network "sees".

4. Conclusions

The modified SSD model proposed in the paper involved developing a technique to identify insects and pests using RGB images. Each network layer was detalied. It is obtained good performances considering IoT and ACC. Also, it is developed a method to visualize what the model has learned. As further work we intend to combine this neural networks with other performing neural networks to obtain a more performant system.

Acknowledgment

This work was supported by a grant of the Ministry of Research, Innovation and Digitization, CNCS/CCCDI – UEFISCDI, project number 202/2020, within PNCDI III.

R E F E R E N C E S

[1] Wen, C., Guyer, D., 2012. Image-based orchard insect automated identification and classification method. Comput. Electron. Agric. 89, 110–115. https://doi.org/10. 1016/j.compag.2012.08.008.

[2] Stoetzel, M.B., 1987. Information on and identification of Diuraphis noxia (Homoptera: Aphididae) and other aphid species colonizing leaves of wheat and barley in the United States. J. Econ. Entomol. 80 (3), 696–704. https://doi.org/10.1093/jee/80.3. 696.

[3] Lecun, Y.; Bengio, Y.; Hinton, G. Deep learning. Nature 2015, 521, 436.

[4] Ciresan D., Meier U., Masci J., Schmidhuber J. 2011. A committee of neural networks for traffic sign classification. The 2011 International Joint Conference on Neural Networks. San Jose: IEEE. 1918–1921.

[5] L., Wang, R., Xie, C., Yang, P., Wang, F., Sudirman, S., Liu, W., 2019. Pestnet: An end-to-end deep learning approach for large-scale multi-class pest detection and classification. IEEE Access 7, 45301–45312.

[6] Ren S, He K, Girshick R, Sun J (2017) Faster R-CNN: towards real-time object detection with region proposal networks. In: IEEE transactions on pattern analysis and machine intelligence.

[7] Redmon J, Farhadi A (2016) YOLO9000: better, faster, stronger. In: IEEE conference on computer vision and pattern recognition.

[8] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C., et al. (2016).SSD: Single Shot MultiBox Detector, in: Proceedings of the European Conference on Computer Vision—ECCV, Amsterdam, The Netherlands, 8–16 October. pp. 21–37.

[9] Qayyum, A., Malik, A. S., Saad, N. M., Iqbal, M., Faris Abdullah, M., Rasheed, W., Rashid Abdullah, T. A., Bin Jafaar, M. Y. (2017), Scene classification for aerial images based on CNN using sparse coding technique. International Journal of Remote Sensing 38(8–10), 2662–2685.

[10] M. Valan, K. Makonyi, A. Maki, D. V., F. Ronqusit (2019), Automated Taxonomic Identification of Insects with Expert-Level Accuracy Using Effective Feature Transfer from Convolutional Networks. -------> [Val 19]

[11] Wright, L.C., Cone, W.W., 1988. Population dynamics of Brachycorynella asparagi (Homoptera: Aphididae) on undisturbed asparagus in Washington State. Environ. Entomol. 17 (5), 878–886. https://doi.org/10.1093/ee/17.5.878.

[12] Espinoza, K., Valera, D.L., Torres, J.A., López, A., Molina-Aiz, F.D., 2016. Combination of image processing and artificial neural networks as a novel approach for the identi fication of Bemisia tabaci and Frankliniella occidentalis on sticky traps in greenhouse agriculture. Comput. Electron. Agric. 127, 495–505. https://doi.org/10.1016/j. compag.2016.07.008.