THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS INC.

SIGNAL PROCESSING
ALGORITHMS, ARCHITECTURES, ARRANGEMENTS, AND APPLICATIONS
SPA 2019
September 18th - 20th, 2019, Poznań, POLAND

Deep interpretable architecture for plant diseases classification

Mohammed Brahimi[1,2,3], Saïd Mahmoudi[2], Kamel Boukhalfa[1], and Abdelouhab Moussaoui[4]

[1]Computer Science Department, USTHB University, Algiers, Algeria
Email: kboukhalfa@usthb.dz
[2]Computer science department, Faculty of engineering, University of Mons, Belgium
Email: Said.MAHMOUDI@umons.ac.be
[3]Computer Science Department, Mohamed El Bachir El Ibrahimi University, Bordj Bou Arreridj, Algeria
Email: mohamed.brahimi@univ-bba.dz
[4]Department of Computer Science, Setif 1 University, Setif, Algeria
Email: moussaoui.abdel@gmail.com

***Abstract*—Recently, many works have been inspired by the success of deep learning in computer vision for plant diseases classification. Unfortunately, these end-to-end deep classifiers lack transparency which can limit their adoption in practice. In this paper, we propose a new trainable visualization method for plant diseases classification based on a Convolutional Neural Network (CNN) architecture composed of two deep classifiers. The first one is named Teacher and the second one Student. This architecture leverages the multitask learning to train the Teacher and the Student jointly. Then, the communicated representation between the Teacher and the Student is used as a proxy to visualize the most important image regions for classification. This new architecture produces sharper visualization than the existing methods in plant diseases context. The proposed visualization method is compared quantitatively with the state-of-the-art methods using Area Over perturbation curve ($AOPC$). The obtained results shows that the proposed visualization method outperforms the existing methods with $AOPC = 0.907$. All experiments are achieved on PlantVillage dataset that contains 54306 plant images.**

***Index Terms*—Plant diseases classification, Deep visualization algorithms, Convolutional Neural Networks (CNNs)**

I. Introduction

Plant diseases cause great damages to agriculture crops by significantly decreasing production [1]. Protecting plants from diseases is vital to guarantee the quality and the quantity of crops [2]. A successful protection strategy starts with an early detection of the disease and the right treatment to prevent its spreading. Many studies proposed the use of Convolutional Neural Network (CNN) to detect and classify diseases. This new trend produced more accurate classifiers compared to shallow machine learning approaches based on hand crafted features [2]–[5]. Despite all these successes, CNN still suffers from the lack of transparency that limits its spreading in many domains. These CNNs are complex deep models that yield good results at the expense of explainability and interpretability. High accuracy is not sufficient for plant diseases classification. Users also need to be informed how the detection is achieved and which symptoms are present in the plant.

In this paper, we propose a classification and visualization architecture based on multitask learning of two classifiers: Teacher and Student. This architecture represents a trainable visualization method for plant diseases classification. The main contribution is the design of an interpretable deep architecture able to do classification and visualization simultaneously. The visualization algorithm is embedded directly in the network design instead of using it after the training as post treatment.

II. Related works

Visualization algorithms are used to explain CNN decision using a heatmap. This heatmap highlights the importance of each pixel for the classifier's decision. These methods backpropagate to the input image the discriminant features used by the network for classification. Most of these methods use heuristics rules during the backpropagation to discard some propagated values. For example, the gradients-based methods like deconvolution [6] and guided backpropagation [7] discard negative gradient values during the backward pass to enhance the quality of visualizations. Furthermore, Layer Wise Relevance Propagation (LRP) methods [8] [9] are based on choosing the layers propagation rules for each layer in the network. These heuristic rules backward the contributions from the output of the network to the input. The global visualization method GRAD-CAM [10] projects the gradient of the last convolution layer to the input image. This gradient is averaged across all the features maps, then resized to the size of the input image using the linear interpolation. The linear interpolation replaces the usual backward pass of the gradient which produces not fine-grained visualization.

These visualization algorithms can produce different heatmaps according to the chosen heuristics and propagation rules, which makes the understanding of the classifier difficult. In the present paper, we combine two classifiers to extract discriminant features from images. These discriminant features

are extracted by the first classifier and projected as an input for the second classifier. This trainable visualization is similar to segmentation architectures like U-Net [11] where the supervision masks are replaced with a second classifier.

III. Proposed Method

We propose a classification and visualization generic architecture, named **Teacher/Student** architecture, based on learning transfer from a first classifier (Teacher) to a second one (Student). This learning transfer from the Teacher to the Student is achieved using an autoencoder having the Teacher as an encoder. The decoder consumes the Teacher latent representations to reconstruct an image with the same dimension of the input image. This image is used as an input of the Student. The whole network (Teacher + Decoder + Student) is trained to minimize jointly the losses of the two classifiers (Teacher and Student). More formally, the network has two outputs YT (Teacher output) and YS (Student output). This outputs YT and YS are Softmax vectors indicating the class of the image. During the training, the loss function (1) is minimised. The hyperparameter α represents the tradeoff between the Teacher loss (2) and the Student loss (3).

$$Loss = \alpha * LossTeacher + (1 - \alpha) * LossStudent \quad (1)$$

$$LossTeacher = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} Y_j^i log(YT_j^i) \quad (2)$$

$$LossStudent = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} Y_j^i log(YS_j^i) \quad (3)$$

The Teacher/Student network is designed to reconstruct an image containing the discriminant features formed by the Teacher to help the Student training. As a side effect of this design, the reconstructed image can be used as a visualization of the important regions for the classification. This architecture represents an autoencoder to denoise the image from irrelevant features from the classification viewpoint. The difference between the usual denoising autoencoder and this architecture lies in the loss function design. The denoising autoencoder minimizes a reconstruction loss while this architecture minimizes the classification loss of two classifiers. This proposed architecture is also designed to extract the important regions for classification without using masks. For this reason, any segmentation architecture can be modified to fit the proposed architecture by modifying the loss function to include a Student classifier and avoid the segmentation masks.

Fig. 1 details the Teacher/Student architecture. For the sake of simplicity, VGG16 [12] architecture is used as Teacher and Student. Nevertheless, the Teacher/Student architecture is flexible and other classification architectures can be used as Teacher or Student. To use another architecture, the decoder must be adapted to inverse the Teacher's layers to reconstruct the input of the Student.

The Teacher/Student architecture is composed of the following components:

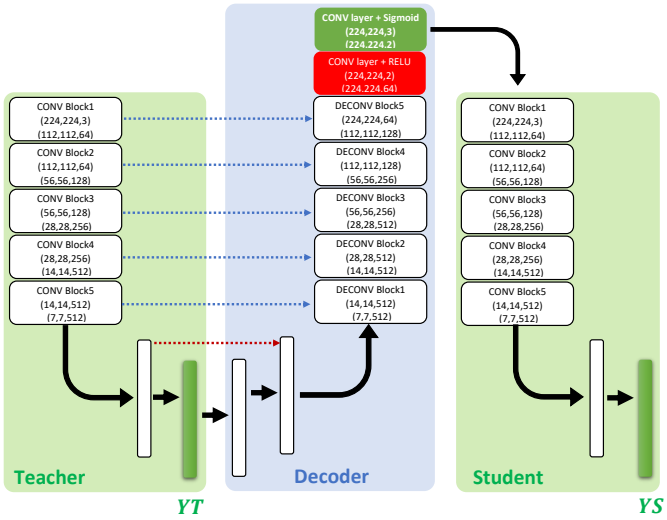

Fig. 1 Teacher/Student network architecture.

TABLE I Deconvolution blocks details.

Layer	Input tensor	Output tensor
Upsampling2d	(x, y, z)	$(2x, 2y, z)$
Conv2D	$(2x, 2y, z)$	$(2x, 2y, z_1)$
Concatenate	$(2x, 2y, z_1) + (2x, 2y, z_1)$	$(2x, 2y, 2z_1)$
Conv2D	$(2x, 2y, 2z_1)$	$(2x, 2y, z_1)$
Conv2D	$(2x, 2y, z_1)$	$(2x, 2y, z_1)$

A. Teacher/Student architecture

The Teacher and the Student architectures are identical to standard architecture VGG16 [12]. Skip connections (blue arrows) are used from the Teacher to the decoder. This skip connections concatenate the input tensors of pooling layer of each convolution block with deconvolution block tensors.

B. Reversed fully connected layers

Convolutional autoencoders reverse only convolution layers to do reconstruction task. Here, the decoder requires discriminant features from fully connected layers. Therefore, two fully connected layers are used to reverse the Teacher's fully connected layers. Furthermore, a skip connection (red arrow) is used to reinforce the decoder by adding the vector of the Teacher's first fully connected layer.

C. Deconvolution blocks

Deconvolution blocks reverse the flow of tensors to form the reconstructed image. The details of this block are shown in Tab. I. This block consumes an input tensor of dimension (x, y, z) and produce a tensor of dimension $(2x, 2y, z_1)$. In each deconvolution block, the input tensor is upsampled, concatenated with the corresponding tensor of skip connection and the depth of resulted tensor after concatenation is reduced. To double the spatial size, deconvolution block uses $Upsampling2d$ layer that repeats the rows and the columns of the tensor by two. Afterwards, Convolution layer is applied to enhance this simple upsampling. The first two deconvolution block (DECONV Block1, DECONV Block2) double the input

tensor spatially without changing its depth ($z_1 = z$). However, the other deconvolution blocks double the tensor spatially and reduce its depth ($z_1 = z/2$).

D. Reconstructed image refinement

After many stages of deconvolution, the spatial dimension of the produced tensor will be equal to input image spatial dimension. However, the depth of the tensor must be reduced to match the depth of the input image. To do this, two convolution layers are applied. The first one (red in Fig. 1) reduces the depth to two channels which enforces the decoder to throw the unnecessary details. The second convolution layer (green in Fig. 1) expands the tensor to three channels to use it as the Student's input. The last decoder's convolution layer uses sigmoid activation function to scale the values in the interval $[0, 1]$. This final tensor is used as Student's input and can be used also as proxy to understand the communication between the Teacher and the Student.

IV. Experimental results

Experimental results are conducted using the segmented version of PlantVillage dataset with black backgrounds [13]. This dataset includes 54306 images of 14 crop species with 38 classes of diseases and healthy plants. The data set is split into a training set that contains 32572 images and a validation set that contains 21734 images. Fig.2 shows the distribution of images in the the classes. In this dataset, three classes contain a high number of images more than 5000 images. However, the rest of the classes contain less than 2500 images. The code source of the proposed visualization method and the comparisons results are available at *https://github.com/Tahedi1/Teacher_Student_Architecture*.

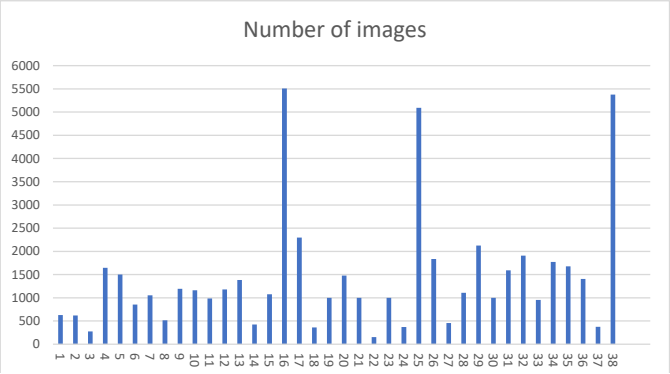

Fig. 2 Distribution of images in the dataset.

A. Classification results

In this section, we present the results of training and validation of the proposed architecture. The training is based on gradient descent algorithm with the following hyperparameters: (learning rate $= 1e{-}4$, momentum $= 0.9$, batch size= 16, number of epochs $= 15$, multitask hyperparamater $\alpha = 0.4$). Training and validation are executed on a workstation containing Graphical Processing Unit GPU Nvidia GTX 1080.

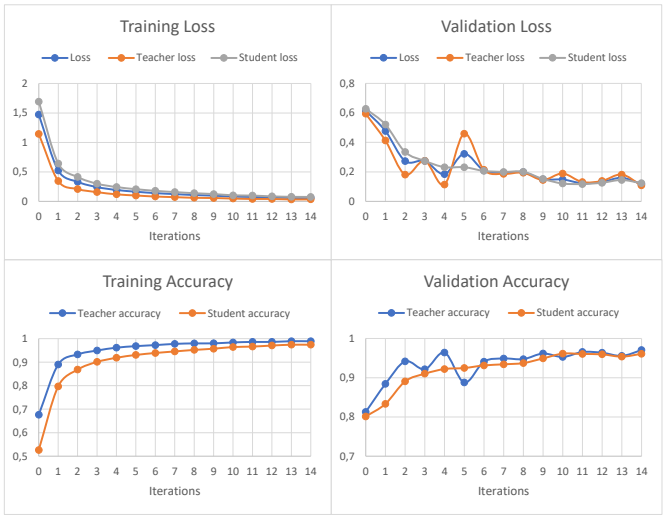

Fig. 3 Classification results of the Teacher/Student architecture.

Fig. 3 shows the results of the training of Teacher/Student architecture. At the beginning of training, the Teacher is more accurate than the student. This may be explained by the dependency of the Student on the representation constructed by the Teacher. However, the loss and the accuracy of the Teacher and the Student converge at the end of the training because the communicated representation becomes stable. Furthermore, the Student's loss is more stable than the Teacher's loss in validation which reinforce the hypothesis of the transfer learning from the Teacher to the Student. This transfer learning is achieved through the quality of the reconstructed image. This reconstructed image focuses on discriminant regions and filters non-discriminant regions. To assess this information filtering mechanism, we analyze the architecture as a visualization method in the following sections.

B. Visualization results

Fig. 4 Visualization images of the Teacher/Student architecture.

The visualizations depicted on Fig. 4 represent the reconstructed three channels images used as an input for the

Student. In Fig. 5, important regions are segmented using a simple binary thresholding algorithm (threshold = 0.9). The thresholding algorithm is applied after a simple aggregation across channels of reconstructed image (noted V) to produce one channel heatmap. To produce this heatmap from the reconstructed image V, the formula (4) is applied. This formula measures the distance between pixel's color and black $(0, 0, 0)$.

$$Heatmap(i, j) = \sqrt{V(i, j, 0)^2 + V(i, j, 1)^2 + V(i, j, 2)^2} \quad (4)$$

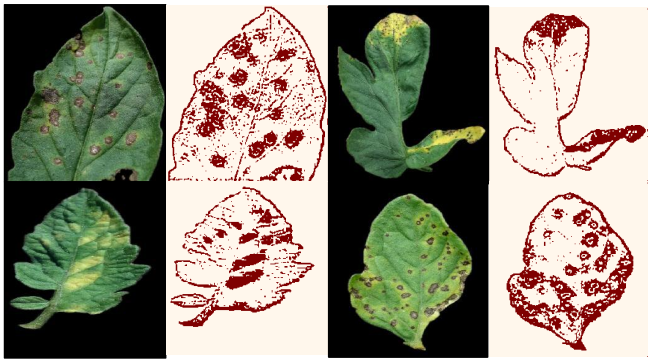

Fig. 5 Heatmaps after threshoulding the visualizations images.

The produced heatmaps (Fig. 5) show clearly the symptoms of the plant disease. Furthermore, these heatmaps are sharp and precise. The healthy regions of the leaves are filtered and only the important regions are highlighted. In the next section, the proposed method is compared quantitatively to other methods using perturbation curves to show its effectiveness.

C. Comparison with visualization algorithms

In this section, the proposed method is compared to the following visualization algorithms : visualization based on gradient [14], Grad-CAM [10] and Layer-wise Relevance Propagation (LRP) methods (Deep Taylor [9], LRP-Epsilon [8], LRP-Z [8]).

1) Heatmaps comparison: Fig. 6 shows the difference between the heatmaps of the visualization algorithms. The proposed algorithm's heatmaps are sharper than the other heatmaps. Indeed, gradient, LRP-Z and LRP-Epsilon heatmaps are noisy and difficult to explain. The gradient heatmaps are noisy because the gradients measure the pixel's sensitivities instead of their contributions. Beside, the presence of negative and positive contributions in LRP heatmaps makes them difficult to understand. On the other hand, Deep Taylor algorithm has good and clear heatmaps compared to other algorithms. The heatmaps of Deep Taylor algorithm highlight almost the same highlighted regions by our algorithm, but with some activated regions on the background. In the proposed algorithm, the background is completely deactivated which gives clean heatmaps.

Fig. 7 shows the difference between the proposed method and Grad-CAM. The Grad-cam algorithm localizes globally the important regions. Furthermore, the Grad-CAM visualizations miss some important regions highlighted by the proposed

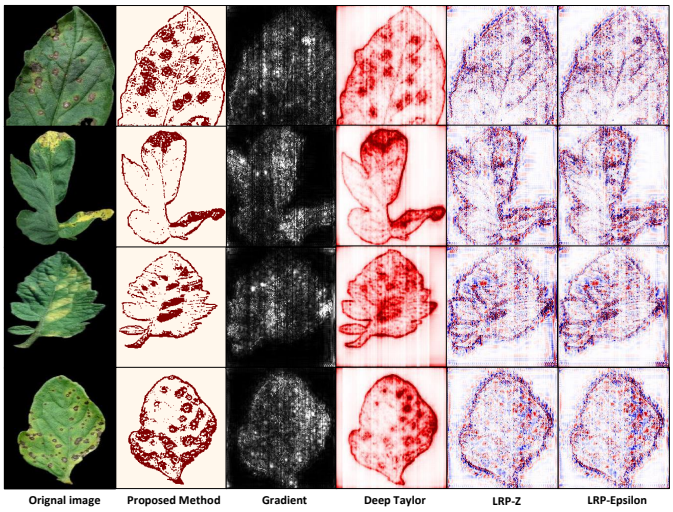

Fig. 6 Comparison with visualization algorithms.

method. This inaccurate localization is due to the resizing used by Grad-CAM to propagate the contributions from the last convolution layer to the input image. In the Teacher/Student architecture, this propagation is ensured by a trainable decoder which makes the visualizations more precise.

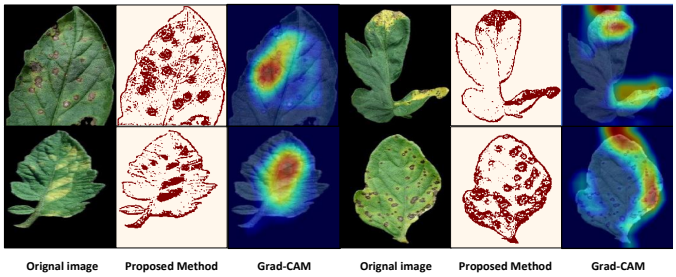

Fig. 7 Comparison with Grad-CAM.

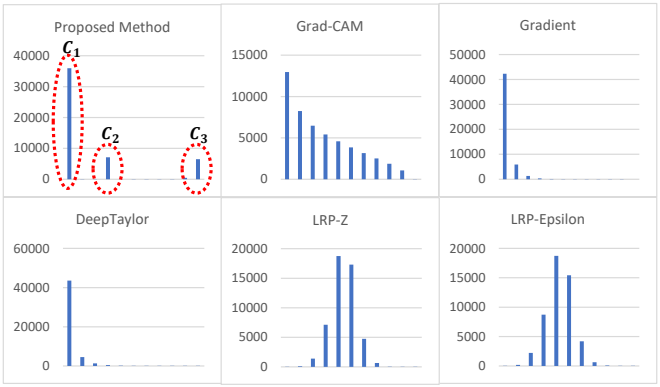

Fig. 8 Histograms of the visualization algorithms. Heatmap values are normalized between 0 and 1. The size of each histogram bin is $\frac{1}{10}$.

2) Histograms of heatmaps values: all the produced heatmaps are normalized in the interval $[0, 1]$ to analyze the

distribution of values in each method. This normalization based on equation (5) where Min and Max are the minimum and the maximum of $Heatmap$ respectively.

$$Heatmap(i,j) = \frac{Heatmap(i,j) - Min}{Max - Min} \tag{5}$$

Histograms of methods are shown in Fig. 8. We notice that the heatmaps values distribution of the proposed method is different from the other methods. Grad-CAM, gradient and Deep Taylor distributions are concentrated in very small values while the density decreases gradually as values increase. LRP-Epsilon and LRP-Z have gaussian-like distribution centered on 0.5. In contrast, the proposed method distribution has three clusters:

- Cluster1 (C_1): 72% of deactivated pixels where the heatmap values are zeros ($Heatmap(pixel) = 0$). This cluster represents the background and non-discriminant pixels.
- Cluster2 (C_2): 15% of pixels having small heatmap's values ($0.2 < Heatmap(pixel) \leq 0.3$).
- Cluster3 (C_3): 13% of pixels having high heatmap values ($0.9 < Heatmap(pixel) \leq 1$) and can be considered as important pixels for the classifier.

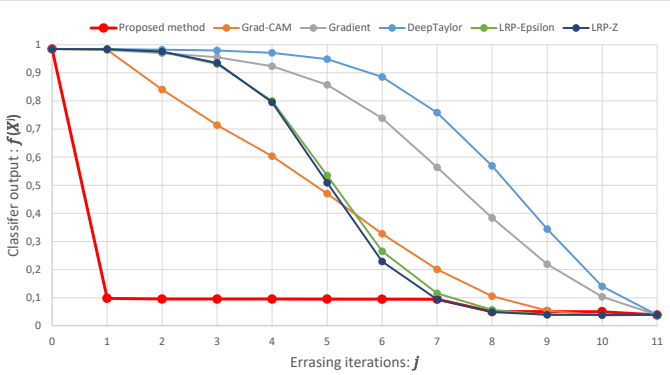

Fig. 9 Perturbation curves of the visualization algorithms.

3) Perturbation curves: to measure the quality of visualization method quantitatively, the produced heatmap is considered as a ranking function of pixels. Good heatmap ranks the pixels correctly according to their importance for the classification. Therefore, if we start erasing the pixels having high values in the heatmap then the classifier output decreases rapidly. To evaluate this ranking, the heatmap values are discretized into intervals of size $\frac{1}{10}$. Afterwards, pixels are erased iteratively according to their heatmaps values in descending order. This erasing procedure is formulated in Algorithm. 1. To erase one pixel, a small black square with a size of three pixels centered on the pixel of interest is used. All dataset images have a black background which motivates the use of black color as reference to erase with. The function $f(X^j)$ gives the classifier estimation of the probability that X^j has the same class of initial image $X^0 = X$. The perturbation curve is traced using points $(j, f(X^{(j)}))$ to track the evolution

Algorithm 1: Perturbation curve experiment for one image X.

Input: X : Input image,
$Heatmap$: Heatmap of input image X
Output: PC : Points list of the perturbation curve,
$AOCP$: Area over perturbation curve

$X^0 \leftarrow X$
$PC.Append((0, f(X^0)))$
$AOPC \leftarrow 0$
$B \leftarrow 1$
$j \leftarrow 1$
for $j \leftarrow 1$ **to** 11 **do**
  if $j < 11$ **then**
    $R \leftarrow \{pixel : B - \frac{1}{10} < Heatmap(pixel) \leq B\}$
  else
    $R \leftarrow \{pixel : Heatmap(pixel) = 0\}$
  $X^j \leftarrow Erase(X^{j-1}, R)$
  $PC.Append((j, f(X^j)))$
  $AOPC \leftarrow AOPC + (f(X^0) - f(X^j))$
  $B \leftarrow B - \frac{1}{10}$
$AOPC \leftarrow \frac{AOPC}{11}$
return $PC, AOPC$

of the classifier output during the erasing. To produce this perturbation curve that characterizes a visualization method, the perturbation curves of validation images are averaged.

Fig. 9 shows the perturbation curves of tested methods. The proposed method's curve decreases rapidly after erasing the pixels of the cluster C_3. Afterwards, before erasing pixels of the cluster C_2 the curve is stationary. Erasing pixels of the cluster C_2 decreases slightly f. Cluster C_1 does not contribute to the decreasing of the classifier output because this cluster contains background and non-discriminant pixels.

Fig. 9 shows that the perturbation curves of the other methods decrease gradually because of the distribution of values in their heatmaps. On the other hand, the perturbation curve of the proposed method decreases steeply because the pixels of C_3 contains the most important pixels in respect to the classifier.

TABLE II Area Over the Perturbation Curve ($AOPC$).

Proposed method	**0,907**
Grad-CAM	0,587
Gradient	0,372
Deep Taylor	0,294
LRP-Z	0,558
LRP-Epsilon	0,550

Tab. II shows Area Over the Perturbation Curve ($AOPC$) for each method. This quantity measures the decreases of the curve compared to its first value formulated in (6) and Algorithm. 1.

$$AOPC = \frac{1}{11}\sum_{i=1}^{N}\sum_{j=1}^{11}\left(f(X_i) - f(X_i^{(j)})\right) \tag{6}$$

The proposed method has better $AOPC = 0.907$ than other methods $AOPC < 0.6$ because of the concentration of important pixels in C_3. In the proposed method, erasing only 13% of image can decrease of f to 0.1.

V. Conclusion and further research

In this work, we have proposed an interpretable Student/Teacher architecture for plant diseases classification. This architecture leverages the multitask to produce a trainable visualization method. Our experiments demonstrate the benefit of adding the Student classifier to guide the architecture in order to reconstruct a sharp visualization images. This reconstructed images contain the discriminant regions, which helps to explain the classifier's decision.

In the future, our objective is to test the Student/Teacher architecture on other classification problems. Besides, we will work to optimize the computation cost of this architecture.

References

[1] I. M. Hanssen and M. Lapidot, "Chapter 2 - major tomato viruses in the mediterranean basin," in *Viruses and Virus Diseases of Vegetables in the Mediterranean Basin*, ser. Advances in Virus Research, G. Loebenstein and H. Lecoq, Eds. Academic Press, 2012, vol. 84, pp. 31 – 66.

[2] M. Brahimi, K. Boukhalfa, and A. Moussaoui, "Deep learning for tomato diseases: Classification and symptoms visualization," *Appl. Artif. Intell.*, vol. 31, no. 4, pp. 299–315, Apr. 2017.

[3] E. Fujita, Y. Kawasaki, H. Uga, S. Kagiwada, and H. Iyatomi, "Basic investigation on a robust and practical plant diagnostic system," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec 2016, pp. 989–992.

[4] Y. Kawasaki, H. Uga, S. Kagiwada, and H. Iyatomi, "Basic study of automated diagnosis of viral plant diseases using convolutional neural networks," in *Advances in Visual Computing*, G. Bebis, R. Boyle, B. Parvin, D. Koracin, I. Pavlidis, R. Feris, T. McGraw, M. Elendt, R. Kopper, E. Ragan, Z. Ye, and G. Weber, Eds. Cham: Springer International Publishing, 2015, pp. 638–645.

[5] L. G. Nachtigall, R. M. Araujo, and G. R. Nachtigall, "Classification of apple tree disorders using convolutional neural networks," in *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, Nov 2016, pp. 472–476.

[6] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *CoRR*, vol. abs/1311.2901, 2013.

[7] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller, "Striving for simplicity: The all convolutional net," in *ICLR (Workshop)*, 2015.

[8] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Mller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLOS ONE*, vol. 10, no. 7, pp. 1–46, 07 2015.

[9] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Mller, "Explaining nonlinear classification decisions with deep taylor decomposition," *Pattern Recognition*, vol. 65, pp. 211 – 222, 2017.

[10] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 618–626.

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[13] D. P. Hughes and M. Salathé, "An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing," *CoRR*, vol. abs/1511.08060, 2015.

[14] K. Simonyan:2014, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *ICLR (Workshop)*, 2014.