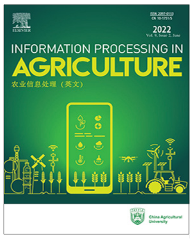
Available at www.sciencedirect.com

INFORMATION PROCESSING IN AGRICULTURE 9 (2022) 212–223

journal homepage: www.elsevier.com/locate/inpa



ResTS: Residual Deep interpretable architecture for plant disease detection

Dhruvil Shah [*], Vishvesh Trivedi, Vinay Sheth, Aakash Shah, Uttam Chauhan

Department of Computer Engineering, Vishwakarma Government Engineering College, Chandkheda, Ahmedabad 382424, Gujarat, India

A R T I C L E I N F O

Article history:
Received 2 January 2021
Received in revised form
25 May 2021
Accepted 4 June 2021
Available online 9 June 2021

Keywords:
Autoencoders
Xception
Deep Learning
Computer Vision
Agriculture

A B S T R A C T

Recently many methods have been induced for plant disease detection by the influence of Deep Neural Networks in Computer Vision. However, the dearth of transparency in these types of research makes their acquisition in the real-world scenario less approving. We propose an architecture named ResTS (Residual Teacher/Student) that can be used as visualization and a classification technique for diagnosis of the plant disease. ResTS is a tertiary adaptation of formerly suggested Teacher/Student architecture. ResTS is grounded on a Convolutional Neural Network (CNN) structure that comprises two classifiers (ResTeacher and ResStudent) and a decoder. This architecture trains both the classifiers in a reciprocal mode and the conveyed representation between ResTeacher and ResStudent is used as a proxy to envision the dominant areas in the image for categorization. The experiments have shown that the proposed structure ResTS (F1 score: 0.991) has surpassed the Teacher/Student architecture (F1 score: 0.972) and can yield finer visualizations of symptoms of the disease. Novel ResTS architecture incorporates the residual connections in all the constituents and it executes batch normalization after each convolution operation which is dissimilar to the formerly proposed Teacher/Student architecture for plant disease diagnosis. Residual connections in ResTS help in preserving the gradients and circumvent the problem of vanishing or exploding gradients. In addition, batch normalization after each convolution operation aids in swift convergence and increased reliability. All test results are attained on the PlantVillage dataset comprising 54 306 images of 14 crop species.

1. Introduction

Agriculture is the primary source of food, raw materials and fuel contributing to a nation's economic growth. As the global population is rising rapidly, agriculture is struggling to meet its needs. Various conditions, including climate change, crop diseases, the decline of pollinators, lack of irrigation, etc., endanger food security. Plant diseases exacerbate the production along with the food quality. To ensure the quality and quantity of crops, it is important to protect plants from diseases [1]. By dramatically reducing the yield, plant diseases inflict severe harm to crops. To ensure food security and the survival of agricultural ecosystems, timely and accurate identification of plant diseases is crucial. One of the most active research fields in agriculture is the early diagnosis of plant diseases [2]. Machine Learning and Computer Vision are being

[*] Corresponding author.
E-mail address: shahdhru2000@gmail.com (D. Shah).

Peer review under responsibility of China Agricultural University.

Abbreviations: DNN, Deep Neural Network; CNN, Convolutional Neural Network; ResTS, Residual Teacher/Student; Conv blocks, Convolution blocks; Deconv blocks, Deconvolution blocks; ReLU, Rectified Linear Unit; GPU, Graphics Processing Unit; CPU, Central Processing Unit; LRP, Layer-wise Relevance Propagation

https://doi.org/10.1016/j.inpa.2021.06.001
2214-3173

fruitful in almost every discipline as they can give more promising outcomes with lower costs. As time advances, more research is being conducted on the applications of these technologies. The agriculture industry is beginning to rely on Deep Learning-based techniques for enhancement in the area [3]. After revolutionizing the domain of Computer Vision, Deep Learning can now solve many tasks such as automated plant disease diagnosis, soil fertility management, rainfall prediction, crop yield prediction, etc. Several Deep Learning-based strategies have been developed for the automatic detection of plant disease to help farmers and increase plant productivity [4–8].

An effective safety policy begins with early identification of the disease and the right medication to deter its spread [9]. The applications of Convolutional Neural Networks (CNNs) to detect and categorize diseases have been indicated in various research. Compared to simplistic Machine Learning methods focused on custom features, this recent trend has created more precise classifiers [4,10–12]. For plant disease classification, high precision is not enough. Users must be apprised of how the diagnosis is done and what manifestations are existing in the particular crop. This can be attained with the help of image processing algorithms so that only the affected regions of a leaf are highlighted [13]. Faster Region-based Convolutional Neural Network (Faster R-CNN), Region-based Fully Convolutional Network (R-FCN) and Single Shot Multibox Detector (SSD) are three families of object-detection algorithms that can be used for localizing the affected areas in the plant [14–15]. For real-time object detection algorithm YOLO (You Only Look Once) is also used in the identification of crop diseases and perceiving the precise locality of the affected regions [16–18]. Despite all these successes, the spread of Convolutional Neural Networks is restricted in many sectors because of their lack of interpretability and explicability. Complex CNN architectures deliver promising outcomes at the cost of transparency.

Various research areas in the agriculture sector that utilizes Deep Learning with Computer Vision have been proven effective. A visualization and classification architecture (Teacher/Student) based on aggregate learning of two deep classifiers was proposed by Brahimi et al. Teacher/Student is a deep interpretable architecture that simultaneously classifies the disease and envisions symptoms [19]. This architecture is a trainable mechanism of visualization for the classification of plant diseases. It uses an autoencoder for preserving the important features in the image. Autoencoder handles the deconstruction and reconstruction of the image so that the noise is removed and only the salient features remain in the image. The teacher is a classifier working as an encoder in the autoencoder. Decoder depletes the inherent representations from the Teacher i.e. the encoder to reconstruct the image. A classifier (Student) learns from this reconstructed image and classifies the disease with finer accuracy. Instead of using it as a post-therapy after preparation, the visualization algorithm is incorporated directly into the Teacher/Student architecture.

Park et al. (2018) investigated that among the several factors which affect crop productivity, crop disease is a critical factor that can cause about a 20–30% reduction in productivity [20]. The inability to take necessary actions in case of infections triggers crop deterioration. Hence, the authors have proposed a mechanism where the farmers will consign the fruit (strawberry) images to the analysis engine. For this purpose, they are using a model based on CNN which would classify the healthy and disease-prone images. The result of the classifier is forwarded to the farmer and the farmer's feedback is reflected in the model. This would improve the accuracy of the classification model. The experiments are performed on a CPU and the model performance is 92% accurate.

As discussed by Jiang et al. (2019), there have been several types of research for the localization of apple leaf diseases that affect the yield of apple fruits, but they lack accurate and fast detection [21]. Thereby, they propose a Deep Learning approach for the real-time detection of apple leaf diseases that is based on improvised CNN architectures. The experiments are conducted on the Apple Leaf Disease Dataset (ALDD) that is prepared from image annotation and data augmentation technologies. It is perceived from the experimental procedures that the proposed INAR-SSD architecture has a detection accuracy of 78.8% and an elevated detection speed of 23.13 Frames per Second. Hence, the authors conclude that the architecture is capable of providing superior accuracy and rapid detection speed than the previous methods for the early detection of apple plant ailments.

It is stated by Montavon et al. (2017) that non-linear techniques such as Deep Neural Networks are suitable for several Machine Learning problems like Image Classification, Natural Language Processing and Human Action Recognition [22]. These techniques function admirably, but one notable drawback of using these methods is that they lack transparency, which limits the interpretability of the solution. Hence, these methods (especially DNNs) act as black boxes. The authors introduce a novel methodology for the clear interpretation of the multi-layer Neural Network by performing Deep Taylor decomposition of the network classification decision. The proposed method is evaluated by conducting the experiments on the MNIST and ILSVRC datasets. From the experimental procedures, the authors deduced that their procedure is stable under heterogeneous architectures and datasets and does not require hyper-parameter tuning.

A deviation from the contemporary approach for object recognition from compact images by using Convolutional Neural Networks is made by Springenberg et al. The general flow is to alter the convolution and max-pool the layers followed by few fully connected layers [23]. However, the authors observed that a replacement of max-pooling can be done by a convolutional layer without any loss of accuracy. Following this idea, they propose a new design composed entirely of convolutional layers and gives up to the mark outcome on object recognition datasets such as ImageNet, CIFAR-10 and CIFAR-100. The authors conclude that uncomplicated architectures may also perform well if modern methods of training CNNs are used. The proposed method of visualization of the representation learned by the higher layers of CNN is uncomplicated and provides sharper visualization of the image regions than the previously recognized tactics.

Several studies that have been induced for utilizing Machine Learning classifiers to shield the plants from diseases by using leaf image processing were analyzed by Brahimi et al. Most of these classifiers are implemented with a compact dataset, focusing on the extraction of indigenous image features for leaf disease classification. The authors use a comparatively large dataset containing 14 828 tomato leaf images infected by 9 diseases for their experiments. They introduce a Convolutional Neural Network for training the classifier which leads to the direct use of the images instead of hand-crafted features. After performing the experiments, the authors have employed visualization techniques to understand the manifestations and locate disease-prone areas in the leaf. The authors infer that the proposed technique outperforms the existing methods, reaching 99.18% of accuracy. Hence, it can be practiced in real-time by the farmers to safeguard the crops against ailments [10].

It is remarked by Bach et al. (2015) that the interpretation of the decisions made by the classifier in automated image classification systems is eminently valuable as it allows the human expert to corroborate the reasoning of the system. Also, it can provide some supplementary knowledge to the human expert. Machine Learning algorithms are triumphant in solving a variety of tasks but they have a drawback of functioning as a black box since they do not provide any information about how they made a certain decision. Hence, the authors propose a solution for understanding the classification decisions by performing pixel-wise decomposition of the non-linear classifiers [24]. The proposed system allows the human expert to visualize the contribution of individual pixels to the prediction in the form of heatmaps. This visualization would allow the expert to verify the truthiness of the classifier decision and provide him with an opportunity to perform extensive analysis on the regions of potential interest. The method is evaluated on classifiers trained on PASCAL VOC 2 009 images, MNIST hand-written digits and the pre-trained ImageNet model. The authors derive that the proposed method is able to produce immensely enlightening heatmaps that reflect in several aspects the sophistication of the learned classifier.

It is stated by Zeiler et al. (2014) that the performance of large convolutional network models on the ImageNet benchmark is quite impressive. However, the reason why they perform so well or how their performance can be improved cannot be clearly understood. The authors propose a novel visualization technique to address these issues. Their technique gives clear insights into the function of the transitional feature layers and the working of the classifier. The authors also study to uncover the contribution of the different model layers to the performance of the classifier. After experimental procedures, the authors draw the inference that the visualizations help in finding architectures that outperform existing techniques on the ImageNet classification standard. Also, they reveal that the ImageNet model is able to generalize to alternative datasets. It outperforms the existing state-of-art results on Caltech-101 and Caltech-256 datasets [25].

The goal of making CNN-based models transparent and easily interpretable was aimed by Selvaraju et al. They proposed a technique called Grad-CAM (Gradient-weighted Class Activation Mapping) for the same [26]. The technique manipulates the gradients of the target concept to produce a coarse localization map that highlights the important regions in the image used for making predictions. They unveil that the approach is applicable for a variety of CNN models. The visualizations leverage more insights into the models and outperform existing methods (on interpretation and faithfulness to original models) in the context of image classification. Further, the authors combined Grad-CAM visualizations with extant high-resolution visualization methods to attain high resolution as well as class-discriminative visualizations. They also devised a scheme to identify principal neurons through Grad-CAM and getting explanations of model decisions. The authors deduce that the proposed method has broad applicability for image classification, image captioning and visual question-answering.

A technique for image-based plant disease detection is proposed by Katafuchi et al. The availability of large datasets encompassing healthy and disease-prone images has led to an increase in the use of Computer Vision techniques for automatic plant disease detection [27]. The authors remark that a deep encoder-decoder network, when trained to reconstruct colors of healthy plant images, fails to reconstruct colors of disease-prone regions. Hence, they focus majorly on the reconstruction phase. They propose a novel framework for plant disease detection which operates a conditional adversarial network- pix2pix. Also, they introduce an anomaly score based on the CIEDE 2 000 color difference. The experiments are performed on the PlantVillage dataset and from the results, it is inferred that the newly proposed system outperforms the pre-existing AnoGAN approach in terms of accuracy and computational efficiency. Hence, the proposed method can be well-used for real-time plant disease detection.

In this research, we carry forward the work done by Brahimi et al. (2019) on Teacher/Student design for plant disease classification [19]. In Teacher/Student, standard VGG16 architecture is employed in the Teacher and the Student components. VGG16 is a linear architecture with no residual connections [28]. Apart from the U-net skip connections in the autoencoder from the encoder to the decoder, there are no residual connections to preserve the gradients in Teacher/Student. We propose an architecture ResTS (Residual Teacher/Student) which uses residual/skip connections in encoder and decoder components. To achieve this, we have implemented standard Xception (Extreme Inception) architecture as Teacher and Student. We have introduced residual/skip connections along with batch normalization in the decoder part of the autoencoder which helps in achieving state-of-the-art accuracy for classification and visualization tasks. Our proposed architecture also contains U-net skip connections in the autoencoder which become beneficial in reconstructing the image [29].

The working of our proposed architecture (ResTS) is elaborated in section 2. Section 3 delineates the results we achieved with our architecture along with the contrast between ResTS and Teacher/Student. Section 4 concludes this paper by outlining various challenges and prospects for the contemporary proposed architecture.

2. Material and methods

2.1. *Material*

For experiments, the PlantVillage dataset is utilized which is an open-source dataset and contains 54 306 images categorized into 38 different classes of infected and healthy crops [13]. These classes represent 14 plant species. ResTS was trained to classify images into these 38 categories and to visualize signs of a specific disease. The segmented version of the dataset that comprises leaf images with black background is used for experiments. The dataset is disbanded into a training set containing 40 729 images, a validation set containing 10 858 images and a test set containing 2 719 images. Depending on the number of images in each class, it is evenly distributed.

2.2. *Proposed method*

We propose a residual Teacher/Student architecture for classification as well as visualization impetus. Fig. 1 demonstrates the ResTS architecture. ResTS structure comprises ResTeacher, Decoder and ResStudent. The general architecture of ResTS is identical to that of Teacher/Student. However, the flow differs as the residual connections are introduced in all the components as shown in Fig. 1. In this architecture, three main tasks are carried out: 1) Deconstruction of the input images by ResTeacher classifier generating intermediate output/latent representations (Out1). 2) Reconstruction of images into original dimensions by the Decoder that consumes the output of ResTeacher. 3) Deconstruction of these replicated images by ResStudent classifier that produces Out2. All three tasks are crucial to visualize the symptoms of the disease and diagnose it.

ResTS uses the same loss functions as the Teacher/Student architecture. ResTS is trained in a way that optimizes loss functions of both the classifiers: ResTeacher and ResStudent. During training, ResTS optimizes the function (1) where *alpha* is the tradeoff between the ResTeacher and ResStudent. The term *alpha* determines how much each classifier's loss function (LossResTeacher and LossResStudent) contributes to the overall loss function. The architecture back-propagates from two different stages (Out1 and Out2) and updates the weights correspondingly [19].

$$Loss = alpha * LossResTeacher + (1 - alpha) * LossResStudent \quad (1)$$

$$LossResTeacher = -\frac{1}{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{C} Y_j^i \log\left(Out1_j^i\right)\right) \quad (2)$$

$$LossResStudent = -\frac{1}{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{C} Y_j^i \log\left(Out2_j^i\right)\right) \quad (3)$$

This architecture is based on learning transfer from one classifier (ResTeacher) to another classifier (ResStudent). ResTeacher and Decoder function as an autoencoder to denoise the images from the classification viewpoint and the ResStudent learns to classify these denoised images. The ResTeacher classifier works as an encoder to reduce the dimensions of the image and the Decoder recreates the image by upscaling the dimensions through a sequence of operations. As a side effect of this unique design, the reconstructed images (output of the Decoder) can be taken as output images for visualizing the features that are dominant in classifying the plant disease. This autoencoder is unlike the usual denoising autoencoders because of the loss function of this architecture. ResTS back-propagates from two points, Out1 and Out2, reducing the classification loss of both classifiers and, as a result, the reconstruction loss of the autoencoder is also minimized.

2.3. *The role of residual connections:*

According to the experiments performed by Kabir et al. (2020), Xception architecture gives state-of-the-art precision when used for multi-plant disease diagnosis [30]. This is because inter-layer outputs are modified by batch normalization into a standard format. For a particular data batch, batch normalization re-calibrates each of the data values depending on the mean and variance. The normalization of batches improves DNN architecture reliability and also contributes to faster convergence. Also, the residual connections sanction the gradients to flow through the architecture without going through non-linear activations. This in turn resolves the problem of vanishing or exploding gradients. Hence, Xception architecture was employed as ResTeacher and ResStudent for classifying the plant disease.

An experiment was performed with standard MobileNetV2 architecture to scrutinize the influence of the residual connections and batch normalization on the reconstruction of images by the Decoder. In this experiment, MobileNetV2 was operated as classifiers instead of Xception architecture as it contains fewer parameters and takes less time in training [31]. Three types of decoders were tested to reconstruct the image from the output of the first classifier: 1) A linear decoder that simply upscales the image dimensions, equivalent to the one used in Teacher/Student architecture. 2) A decoder that recreates the image using residual connections. 3) A decoder that uses residual connections when recreating the image together with batch normalization after each convolution operation.

Fig. 2 shows how a reconstructed image improved over 3 types of the decoder. The linear decoder with no residual connections produced an image that is arduous to comprehend whereas the decoder with residual connections devised better visualizations than the former. However, the decoder with residual connections and batch normalization (after each convolution operation) gave the best images that only contain key regions from the perspective of disease identification. Hence, it is deduced that residual connections along with batch normalization help in reducing the recreation loss of the images and the adoption of residual connections and batch normalization in the Decoder of ResTS are justified. The Decoder in ResTS is built in such a way that it inversely adapts the standard Xception architecture to ensure that the image is reconstructed in the same manner it was decon-

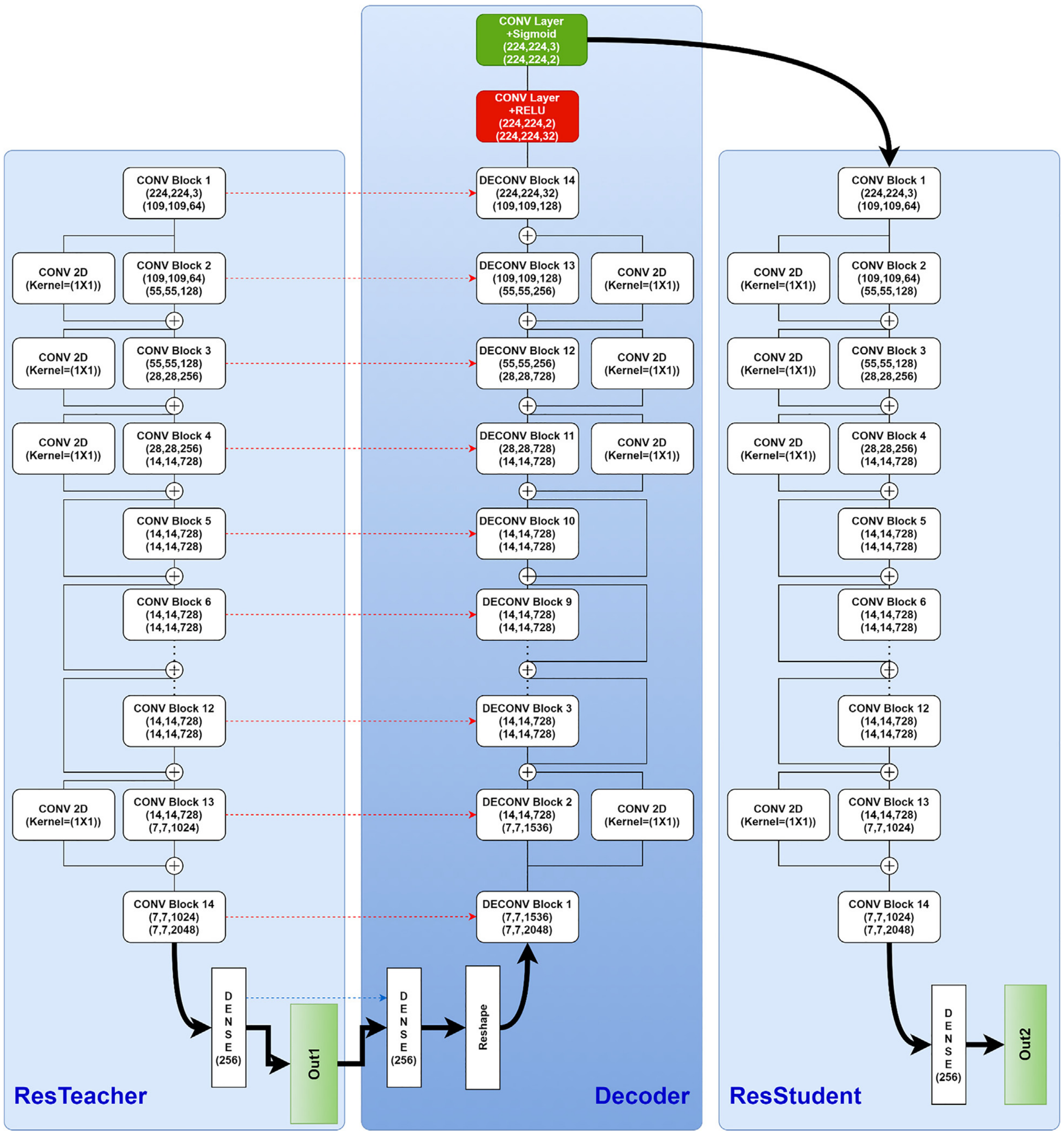

CONV Layer +Sigmoid (224,224,3) (224,224,2)
CONV Layer +RELU (224,224,2) (224,224,32)
CONV Block 1 (224,224,3) (109,109,64)
CONV Block 2 (109,109,64) (55,55,128)
CONV Block 3 (55,55,128) (28,28,256)
CONV Block 4 (28,28,256) (14,14,728)
CONV Block 5 (14,14,728) (14,14,728)
CONV Block 6 (14,14,728) (14,14,728)
CONV Block 12 (14,14,728) (14,14,728)
CONV Block 13 (14,14,728) (7,7,1024)
CONV Block 14 (7,7,1024) (7,7,2048)
CONV 2D (Kernel=(1X1))
DECONV Block 14 (224,224,32) (109,109,128)
DECONV Block 13 (109,109,128) (55,55,256)
DECONV Block 12 (55,55,256) (28,28,728)
DECONV Block 11 (28,28,728) (14,14,728)
DECONV Block 10 (14,14,728) (14,14,728)
DECONV Block 9 (14,14,728) (14,14,728)
DECONV Block 3 (14,14,728) (14,14,728)
DECONV Block 2 (14,14,728) (7,7,1536)
DECONV Block 1 (7,7,1536) (7,7,2048)
DENSE (256)
Out1
Reshape
Out2
ResTeacher
Decoder
ResStudent

Fig. 1 – ResTS network architecture.

structed by ResTeacher. ResTS is composed of the following constituents:

2.4. *ResTeacher and ResStudent*

The ResTeacher and ResStudent architectures are equivalent to the standard Xception architecture. There are 12 residual connections in Xception, four of which go through Conv2D layers with 1 × 1 kernel size. The batch normalization is present after each separable convolutional layer (SeparableConv2D) [32]. ResTeacher and ResStudent use a dense layer of 256 nodes and another dense layer of 38 nodes (Out1) with ‘softmax’ activation to substitute the last fully connected layers in the regular Xception architecture.

2.5. *Decoder*

The U-net skip connections (red arrows) go from ResTeacher to the Decoder as shown in Fig. 1. These connections concatenate the last layer of each Convolution Block (Batch normalization layer or Activation layer) with the first layer of each Deconvolution Block. U-net skip connections aid the Decoder

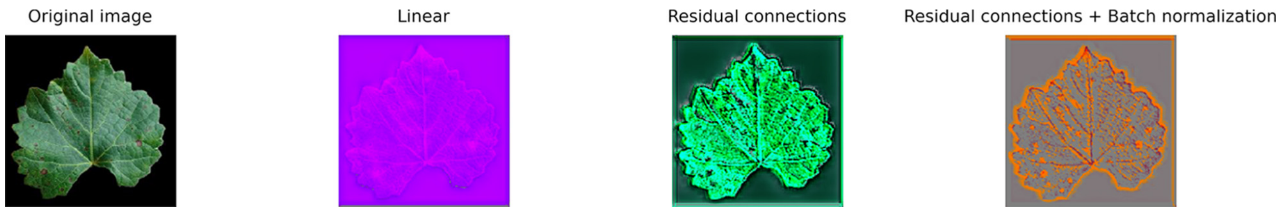


Fig. 2 – Reconstructed images from different types of the decoder according to MobileNetV2.

in maintaining the original image's features when reconstructing it. The Decoder is given 12 residual connections in symmetrical reverse order from the Xception architecture. Furthermore, batch normalization is performed after each separable convolutional layer (SeparableConv2D) in the Decoder which was proven fruitful while reconstructing the images. The dense layer (256 nodes) in the ResTeacher is connected with the dense layer (256 nodes) of the Decoder via a skip connection (blue arrow) which is shown in Fig. 1. This assists the Decoder grasp only the salient features from ResTeacher. This dense layer from the Decoder produces a tensor of shape (256) and it is converted into (7, 7, 2 048) dimensional tensor with the help of a dense layer and a reshaping layer (Reshape) respectively. This tensor is then passed to the DECONV Block 1 for the reconstruction of the image.

2.6. *Working of DECONV blocks*

Xception architecture contains Separable convolution layers instead of regular convolution layers. In the Decoder, there are 14 Deconv blocks and every dimension is expanded step by step in an identical manner in which they were truncated in ResTeacher. Each Deconv block receives skip connections from the ResTeacher. Each Deconv block except for the first and the last one has a residual connection. Table 1 depicts the disposition of the layers of Deconv blocks 1–14 and how the dimensions of the images are upscaled at each stage. The '+' sign denotes addition and the '||' sign denotes the concatenation of tensors in Table 1.

DECONV Blocks reconstruct the image by reversing the flow of tensors. For instance, block 1 in Table 1 consumes

Table 1 – Summary of the Deconvolution blocks in ResTS.

DECONV Blocks	Layers	Input tensor	Output tensor
1	SeparableConv2D	(7, 7, 2 048)	(7, 7, 2 048)
	Concatenate (Skip)	(7, 7, 2 048) || (7, 7, 2 048)	(7, 7, 4 096)
	SeparableConv2D	(7, 7, 4 096)	(7, 7, 1 536)
2	UpSampling2D	(7, 7, 1 536)	(14, 14, 1 536)
	SeparableConv2D	(14, 14, 1 536)	(14, 14, 1 024)
	Concatenate (Skip)	(14, 14, 1 024) || (14, 14, 1 024)	(14, 14, 2 048)
	SeparableConv2D	(14, 14, 2 048)	(14, 14, 728)
	Add (Residual)	(14, 14, 728) + (14, 14, 128)	(14, 14, 728)
3–10	SeparableConv2D	(14, 14, 728)	(14, 14, 728)
	Concatenate (Skip)	(14, 14, 1 456) || (14, 14, 1 456)	(14, 14, 1 456)
	SeparableConv2D	(14, 14, 728)	(14, 14, 728)
	SeparableConv2D	(14, 14, 728)	(14, 14, 728)
	Add (Residual)	(14, 14, 728) + (14, 14, 728)	(14, 14, 728)
11	UpSampling2D	(14, 14, 128)	(28, 28, 728)
	SeparableConv2D	(28, 28, 728)	(28, 28, 728)
	Concatenate (Skip)	(28, 28, 1 456) || (28, 28, 1 456)	(28, 28, 1 456)
	SeparableConv2D	(28, 28, 728)	(28, 28, 728)
	Add (Residual)	(28, 28, 728) + (28, 28, 728)	(28, 28, 728)
12	Conv2Dtranspose (strides= (2x2))	(28, 28, 728)	(57, 57, 1)
	SeparableConv2D (padding = 'valid')	(57, 57, 1)	(55, 55, 256)
	Concatenate (Skip)	(55, 55, 256) || (55, 55, 256)	(55, 55, 512)
	SeparableConv2D	(55, 55, 256)	(55, 55, 256)
	Add (Residual)	(55, 55, 256) + (55, 55, 256)	(55, 55, 256)
13	Conv2DTranspose (strides= (2x2))	(55, 55, 256)	(111, 111, 1)
	SeparableConv2D (padding = 'valid')	(111, 111, 1)	(109, 109, 128)
	Concatenate (Skip)	(109, 109, 128) || (109, 109, 128)	(109, 109, 256)
	SeparableConv2D	(109, 109, 256)	(109, 109, 128)
	Add (Residual)	(109, 109, 128) + (109, 109, 128)	(109, 109, 128)
14	Conv2D	(109, 109, 128)	(109, 109, 64)
	Concatenate (Skip)	(109, 109, 64) || (109, 109, 64)	(109, 109, 128)
	ZeroPadding2D	(109, 109, 128)	(111, 111, 128)
	Conv2D	(111, 111, 128)	(111, 111, 32)
	UpSampling2D	(111, 111, 32)	(222, 222, 32)
	ZeroPadding2D	(222, 222, 32)	(224, 224, 32)

an input tensor of dimensions (7, 7, 2 048). This tensor is passed into the SeparableConv2D layer of 2 048 nodes and then it is concatenated with the tensor of the U-net skip connection of the CONV Block 14 that also possesses the same dimension. Then, it is passed through the SeparableConv2D layer that contains 1 536 nodes resulting in a (7, 7, 1 536) dimensional tensor. Throughout the network, the UpSampling2D layer is utilized to increase the height and width of an image while keeping the number of channels constant. UpSampling2D layer consumes a tensor of dimensions (X, Y, Z) and yields a tensor of (2X, 2Y, Z). Similarly, Add layers (for residual connections) and Concatenate layers (for skip connections) are applied for adding two tensors ((X, Y, Z) + (X, Y, Z)) and appending two tensors ((X, Y, Z) || (X, Y, Z1)) respectively. ZeroPadding2D layer is used to pad the tensor with zeros which turns a tensor of dimensions (X, Y, Z) into a tensor having dimensions (X + 2, Y + 2, Z). Conv2DTranspose layer with strides of (2, 2) is used in block 12 and block 13 to upsample the image to produce certain dimensions.

After Deconv block 14, the dimension of the image is in the form of (224, 224, 32) dimensional tensor that is fed into two convolutional layers with activation functions 'ReLU' and 'Sigmoid' respectively. The first convolutional layer reduces the number of channels from 32 to 2 so that the inessential information is discarded. The second convolutional layer expands the number of channels to 3 while using 'Sigmoid' as an activation function which causes the values to remain in the interval 0 to 1. These two layers forward the images of the dimensions (224, 224, 3) to ResStudent.

3. Results and discussion

3.1. *Classification results of ResTS*

We discuss the outcomes of training and validation of ResTS architecture in this section. ResTS is trained on the dataset for 15 epochs. The optimizer used here is Stochastic Gradient Descent with a learning rate of 0.000 1 and the batch size is set to 16. The tradeoff (alpha) between ResTeacher and ResStudent is set to 0.4. Categorical cross-entropy is used as a loss function for both classifiers as this is a multiclass classification.

Fig. 3 depicts that ResTeacher's accuracy is higher than that of ResStudent's at the beginning. The reliance of the ResStudent on the representation constructed by the ResTeacher may be the explanation for this. At the end of the training, however, the loss and the accuracy of the ResTeacher and the ResStudent coincide as the communicated image among them becomes steadier. The image recreated by the Decoder focuses on areas that are discriminating and filters regions that are not discriminating. We evaluate the architecture as a visualization method in the next section.

3.2. *Visualization results*

Fig. 4 portrays few input images (green leaf images) and the corresponding reconstructed images generated by autoencoder (ResTeacher + Decoder) that are used as an input for ResStudent. These reconstructed images are the output of the Decoder having (224, 224, 3) dimensions and referred to as V.

After basic aggregation through the reconstructed image channels, the thresholding algorithm is implemented to generate one channel heatmap. The formula (4) is added for basic aggregation through the (Red, Green, Blue) channels in the reconstructed picture V. It determines the distance between the color of the pixel and background color (R = 0.149, G = 0.153, B = 0.341) [19]. Blue color has a higher value (0.341) than the other two as it is dominant in V.

$$Heatmap(i,j) = \sqrt{(V(i,j,0) - 0.149)^2 + (V(i,j,1) - 0.153)^2 + (V(i,j,2) - 0.341)^2} \quad (4)$$

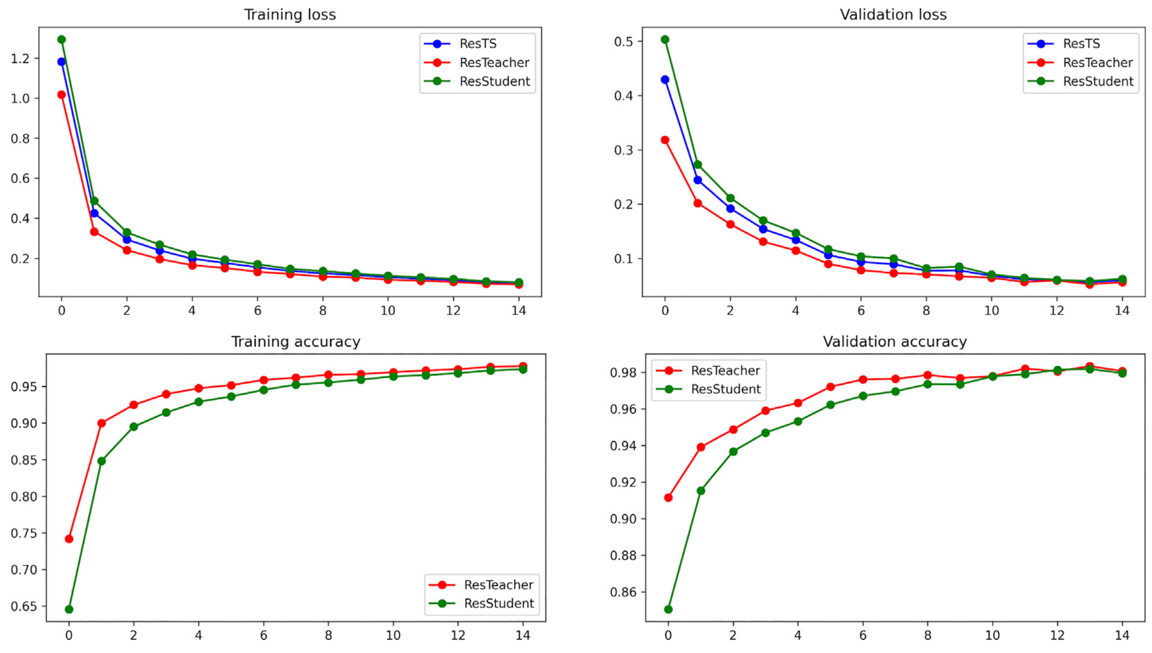


Fig. 3 – Evaluation metrics of ResTS architecture.

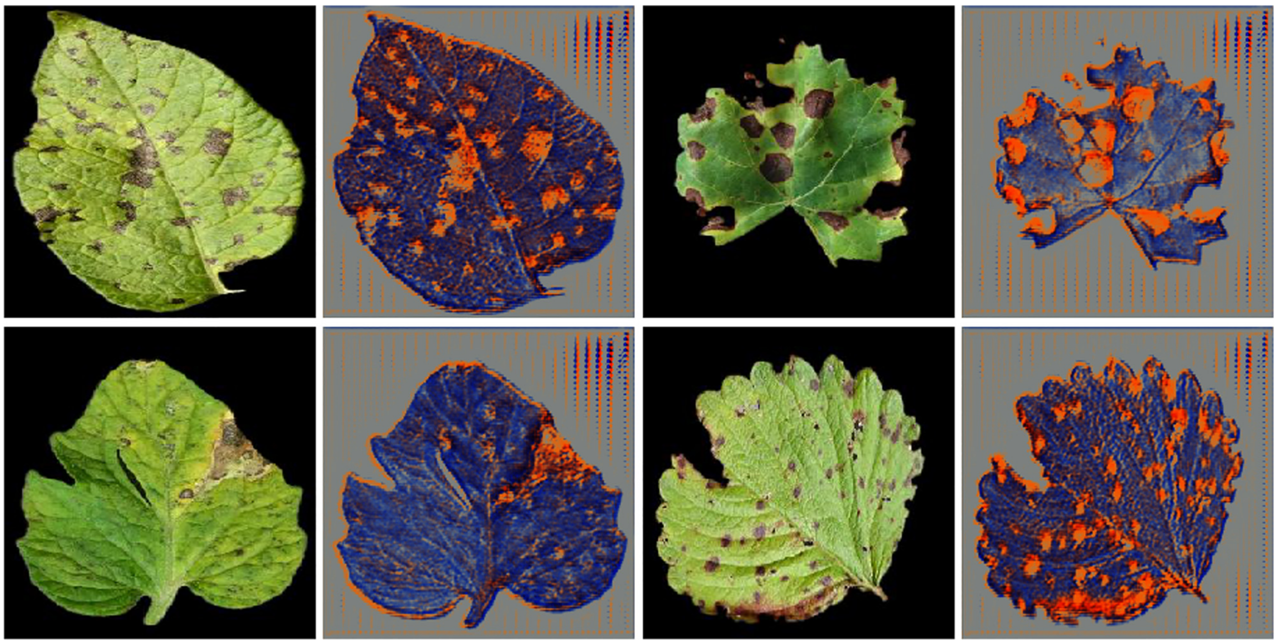


Fig. 4 – ResTS reconstructed images.

A simple binary thresholding algorithm (threshold = 0.6) is applied to segment significant regions from the output of the above formula (noted *Heatmap*). The values greater than the threshold (0.6) are set to 1 for envisioning the disease-specific spots. Fig. 5 portrays the same input images (green leaf images) as Fig. 4 and the corresponding heatmaps after thresholding. These heatmaps picture the manifestations of the specific disease and are sharper and concise. The stable leaf areas are screened and notable regions are emphasized specifically.

The performances of Teacher/Student and ResTS are collated in the following section. Moreover, the visualizations of the proposed method (ResTS) are juxtaposed with those of Teacher/Student architecture's in the subsequent section.

3.3. *Comparison with the Teacher/Student architecture*

The Teacher/Student model is trained with the same hyper-parameters and the same dataset split as ResTS. At the end of 15 epochs, both the architectures present a smooth curve for training loss. Although, the curve for validation loss diverges as Teacher/Student architecture gives a curve with spikes whereas ResTS gives a smoother curve. This is evident in Fig. 6.

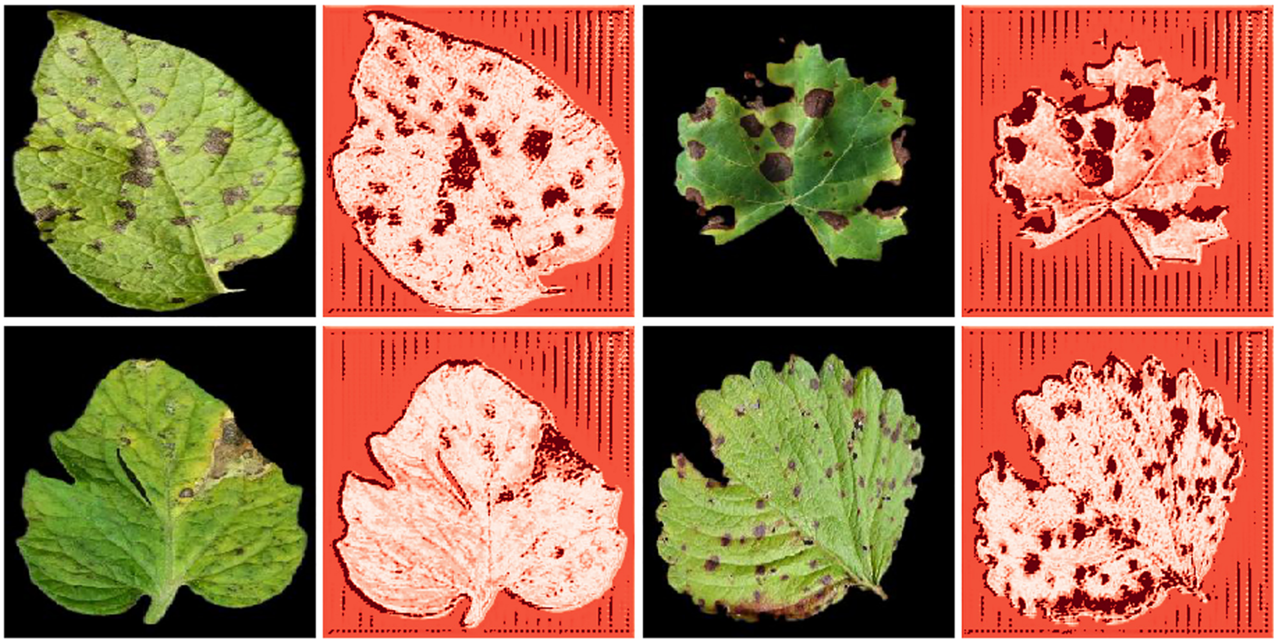


Fig. 5 – ResTS heatmap images.

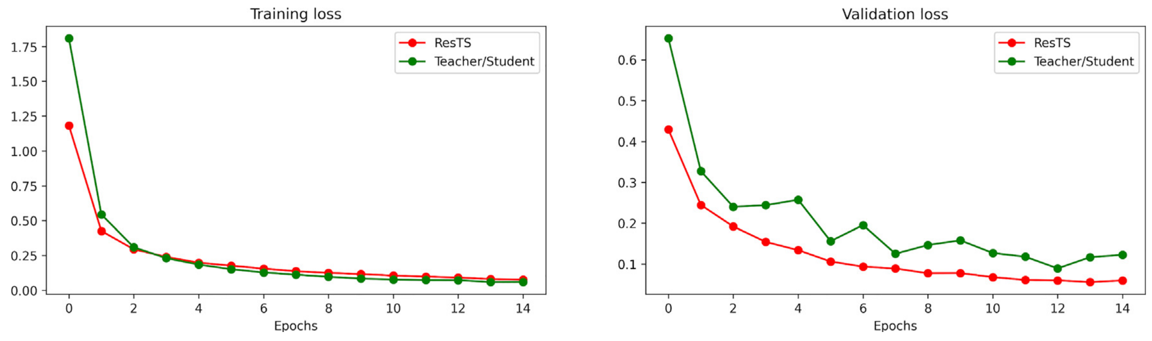


Fig. 6 – Comparison of ResTS and Teacher/Student performance.

The spikes in the validation loss curve of Teacher/Student indicate that the model is significantly overfitting and performing poorly on the validation set compared to the training set. However, the ResTS model is performing well on both training and validation sets. It is palpable that the addition of residual connections and performing batch normalization after each convolution operation in the Decoder assists in converging faster and avoiding overfitting. Table 2 summarizes the evaluation metrics of both models. It is tangible that the Teacher/Student model is leaning towards the training set resulting in an inferior training loss at the expense of excessive validation loss and pruned test accuracy. This means that the Teacher/Student is not able to perform well on images other than the training set i.e. real-world images.

To evaluate the models, 2 719 images were reserved as test images at the time of dataset split. These images were not part of the training and can help assess the models for the real-world scenario. Though accuracy is a good metric for appraising a Machine Learning model, the F1 score is more suitable for evaluation when the class distribution in the dataset is unbalanced as it considers the number of instances of each class. Here, the weighted F1 score was used to account for label imbalance in the dataset. The output of the first classifier (ResTeacher in the case of ResTS) was exploited to calcu-

Table 2 – Performance of ResTS and Teacher/Student in the same environment.

Architecture	Training loss	Validation loss	F1 Score
ResTS	0.077	0.059	0.991
Teacher/Student	0.060	0.123	0.972

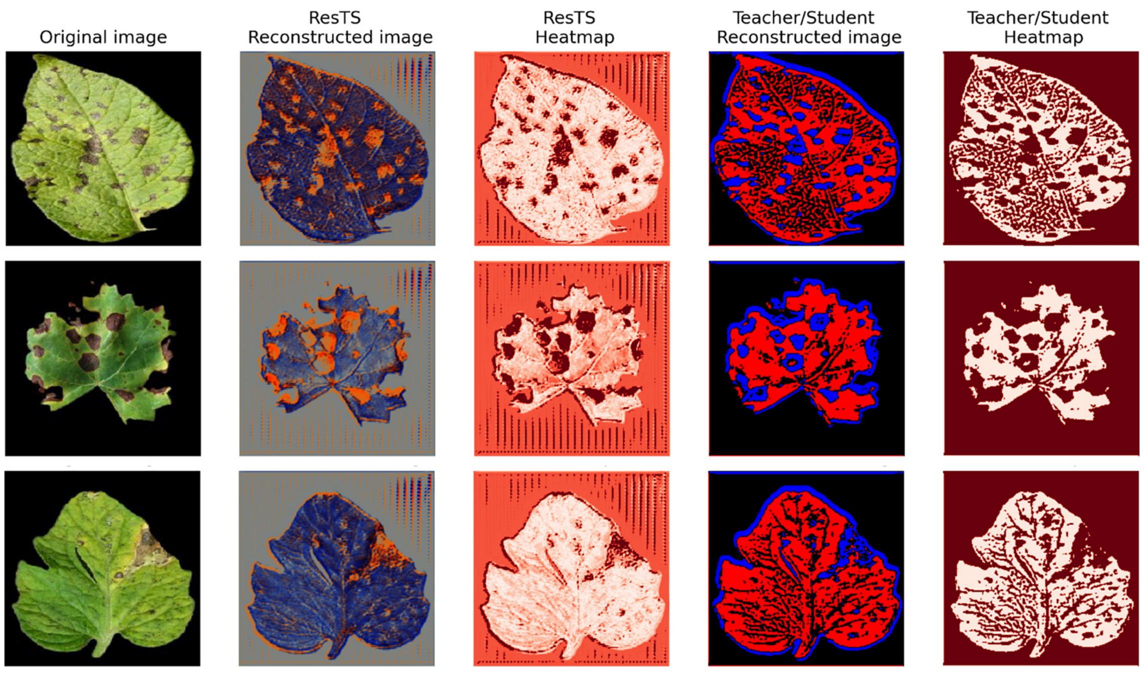

Original image
ResTS Reconstructed image
ResTS Heatmap
Teacher/Student Reconstructed image
Teacher/Student Heatmap

Fig. 7 – Comparison between ResTS and Teacher/Student.

late the F1 score. With an F1 score of 0.990 8, ResTS architecture appears to be functioning well on images it has not encountered before. As a result, ResTS outperforms the preceding Teacher/Student architecture in predicting the correct disease group, making it more reliable in practice.

Furthermore, Fig. 7 shows the comparison between the visualization outputs of ResTS and Teacher/Student. For the creation of a heatmap for Teacher/Student, the same thresholding algorithm was used with the threshold value of 0.6 and the same formula is applied to find the distance between the point (0.9, 0, 0) as the red color is dominant in the reconstructed image of Teacher/Student. The visualizations created by ResTS are more accurate and finer than those of Teacher/Student architecture. The heatmaps by Teacher/Student also activate the features that are not discriminant while heatmaps from ResTS is highlighting only the discriminant features of a particular disease. The time it necessitates ResTS and Teacher/Student architecture to replicate a single image is 143 ms and 115 ms, respectively, while the time entailed estimating the final outputs (Out1 and Out2) is 150 ms and 228 ms.

ResTS (Residual Teacher/Student) architecture includes 153.4 million parameters and consequently higher space requirements while Teacher/Student architecture, which uses a linear VGG16 architecture, comprises 68 million parameters and comparatively, it necessitates lesser memory. When run on a Tesla T4 GPU having 12.72 GB RAM, 1 epoch entails 45 min and 51 min for Teacher/Student and ResTS models respectively, while training.

3.4. *Comparison with other methods*

In this section, the visualizations created by Grad-CAM, Gradient, LRP-Z, LRP-Epsilon methods are discussed [19]. These algorithms create heatmaps that are noisy and uneasy to interpret for visualization of symptoms of the disease [24]. The gradient heatmaps are noisy because, instead of their contributions, the gradients measure the sensitivities of the pixel. In comparison, the inclusion of negative and positive inputs in LRP heatmaps makes it difficult to grasp them [33]. On the other hand, compared with other algorithms, the Deep Taylor algorithm has strong and simple heatmaps but it also activates some background regions. The Grad-CAM algorithm localizes the discriminant regions globally. In addition, some major key regions for disease identification are absent from the Grad-CAM visualizations [23]. ResTS can yield the best visualizations that recognize the plant disease-specific spots.

4. Conclusion

With the cooperation of Deep Learning, the problem of early diagnosis of plant diseases can be resolved and food security can be ensured. Numerous methods have been proposed for the same, making the agriculture sector one of the most active research areas in Machine Learning and Computer Vision disciplines [34–36]. In this research, we have worked on improving the results of formerly proposed Teacher/Student architecture by introducing residual connections and carrying out batch normalization in all three components of it: Teacher, Decoder and Student. ResTS comprises an autoencoder (ResTeacher + Decoder) and ResStudent. ResStudent is guiding the entire architecture to train in a way that the reconstructed images by decoder contain only the discriminant regions. Residual connections and batch normalization are two factors that make ResTS novel. ResTeacher, Decoder and ResStudent contain residual connections to assist the gradients flow throughout the network which resolves the issue of vanishing or exploding gradients. Batch normalization carried out after each convolution operation in ResTS contributes to faster convergence. We trained both the architectures (ResTS and Teacher/Student) on the same dataset in the same environment to collate them. From the results of experiments performed, it was deduced that the Teacher/Student structure that used VGG16 is overfitting to the training set and performs poorly on the validation set (Validation loss: 0.123) while ResTS performs well on the training set as well as the validation set (Validation loss: 0.059). The visualizations from ResTS are precise and superior to those of Teacher/Student. Model's meticulous decisions can be explained by its unique design. In the future, any other residual architecture can be fit to work as the ResTeacher or ResStudent. To use a residual classifier other than Xception, the layers in the Decoder have to be applied in an exact inverse manner as ResTeacher. The residual connections also need to be applied in the opposite order as ResTeacher.

5. Availability of data and material

All relevant data and material are presented in the main paper.

6. Consent for publication

Not applicable.

7. Ethics approval and consent to participate

Not applicable.

Funding

Not Applicable.

Declaration of Competing Interest

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

Acknowledgements

The authors are grateful to Vishwakarma Government Engineering College for the permission to publish this research.

Appendix A. . Dataset and source code access

The PlantVillage dataset used in this research is available at https://github.com/spMohanty/PlantVillage-Dataset/

tree/master/raw/segmented. To reproduce the results portrayed in this paper or to conduct further research, the source code of the proposed architecture can be found at https://github.com/jackfrost1411/Residual_Teacher_Student

REFERENCES

[1] Altieri MA. Agroecology: The science of sustainable agriculture. 2018. https://doi.org/10.1201/9780429495465.

[2] Hasan RI, Yusuf SM, Alzubaidi L. Review of the state of the art of deep learning for plant diseases: A broad analysis and discussion. Plants 2020;9. https://doi.org/10.3390/plants9101302.

[3] Cap QH, Uga H, Kagiwada S, Iyatomi H. LeafGAN: An Effective Data Augmentation Method for Practical Plant Disease Diagnosis. IEEE Trans Autom Sci Eng 2020. https://doi.org/10.1109/TASE.2020.3041499.

[4] Hidayatuloh A, Nursalman M, Nugraha E. Identification of Tomato Plant Diseases by Leaf Image Using Squeezenet Model. 2018 Int. Conf. Inf. Technol. Syst. Innov. ICITSI 2018 - Proc. 2018. https://doi.org/10.1109/ICITSI.2018.8696087.

[5] Howlader MR, Habiba U, Faisal RH, Rahman MM. Automatic Recognition of Guava Leaf Diseases using Deep Convolution Neural Network. 2nd Int. Conf. Electr. Comput. Commun. Eng. ECCE 2019, 2019.. https://doi.org/10.1109/ECACE.2019.8679421.

[6] Kulkarni O. Crop Disease Detection Using Deep Learning. Proc. - 2018 4th Int. Conf. Comput. Commun. Control Autom. ICCUBEA 2018, 2018. https://doi.org/10.1109/ICCUBEA.2018.8697390.

[7] Liu B, Ding Z, Zhang Y, He D, He J. Kiwifruit Leaf Disease Identification Using Improved Deep Convolutional Neural Networks. Proc. - 2020 IEEE 44th Annu. Comput. Software, Appl. Conf. COMPSAC 2020, 2020. https://doi.org/10.1109/COMPSAC48688.2020.00-82.

[8] Saleem MH, Potgieter J, Arif KM. Plant disease detection and classification by deep learning. Plants 2019;8. https://doi.org/10.3390/plants8110468.

[9] Chapaneri R, Desai M, Goyal A, Ghose S, Das S. Plant Disease Detection: A Comprehensive Survey. 2020 3rd Int. Conf. Commun. Syst. Comput. IT Appl. CSCITA 2020 - Proc., 2020. https://doi.org/10.1109/CSCITA47329.2020.9137779.

[10] Brahimi M, Boukhalfa K, Moussaoui A. Deep Learning for Tomato Diseases: Classification and Symptoms Visualization. Appl Artif Intell 2017;31. https://doi.org/10.1080/08839514.2017.1315516.

[11] Fujita E, Kawasaki Y, Uga H, Kagiwada S, Iyatomi H. Basic investigation on a robust and practical plant diagnostic system. Proc. - 2016 15th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2016, 2017. https://doi.org/10.1109/ICMLA.2016.56.

[12] Kawasaki Y, Uga H, Kagiwada S, Iyatomi H. Basic study of automated diagnosis of viral plant diseases using convolutional neural networks. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9475, 2015. https://doi.org/10.1007/978-3-319-27863-6_59.

[13] Hughes David P, Salathé Marcel. An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing. ArXiv 2015.

[14] Fuentes A, Yoon S, Kim SC, Park DS. A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. Sensors (Switzerland) 2017;17. https://doi.org/10.3390/s17092022.

[15] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans Pattern Anal Mach Intell 2017;39. https://doi.org/10.1109/TPAMI.2016.2577031.

[16] Katumba A, Bomera M, Mwikirize C, Namulondo G, Ajero MG, Ramathani I, et al. A Deep Learning-based Detector for Brown Spot Disease in Passion Fruit Plant Leaves. ArXiv 2020.

[17] Liu J, Wang X. Early recognition of tomato gray leaf spot disease based on MobileNetv2-YOLOv3 model. Plant Methods 2020;16. https://doi.org/10.1186/s13007-020-00624-2.

[18] Liu J, Wang X. Tomato Diseases and Pests Detection Based on Improved Yolo V3 Convolutional Neural Network. Front Plant Sci 2020;11. https://doi.org/10.3389/fpls.2020.00898.

[19] Brahimi M, Mahmoudi S, Boukhalfa K, Moussaoui A. Deep interpretable architecture for plant diseases classification. Signal Process. - Algorithms, Archit. Arrange. Appl. Conf. Proceedings, SPA, vol. 2019- September, 2019. https://doi.org/10.23919/SPA.2019.8936759.

[20] Park H, Jeesook E, Kim SH. Crops Disease Diagnosing Using Image-Based Deep Learning Mechanism. Proc. 2nd Int. Conf. Comput. Netw. Commun. CoCoNet 2018, 2018. https://doi.org/10.1109/CoCoNet.2018.8476914.

[21] Jiang P, Chen Y, Liu B, He D, Liang C. Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolutional Neural Networks. IEEE. Access 2019;7. https://doi.org/10.1109/ACCESS.2019.2914929.

[22] Montavon G, Lapuschkin S, Binder A, Samek W, Müller KR. Explaining nonlinear classification decisions with deep Taylor decomposition. Pattern Recognit 2017;65. https://doi.org/10.1016/j.patcog.2016.11.008.

[23] Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M. Striving for simplicity: The all convolutional net. 3rd Int. Conf. Learn. Represent. ICLR 2015 - Work. Track Proc., 2015.

[24] Bach S, Binder A, Montavon G, Klauschen F, Müller KR, Samek W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS ONE 2015;10. https://doi.org/10.1371/journal.pone.0130140.

[25] Zeiler MD, Fergus RLNCS. Visualizing and understanding convolutional networks. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics) 2014;8689. https://doi.org/10.1007/978-3-319-10590-1_53.

[26] Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. Proc. IEEE Int. Conf. Comput. Vis., vol. 2017- October, 2017. https://doi.org/10.1109/ICCV.2017.74.

[27] Katafuchi R, Tokunaga T. Image-based plant disease diagonasis with unsupervised anomaly detection based on reconstructability of colors. ArXiv 2020.

[28] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., 2015.

[29] Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9351, 2015. https://doi.org/10.1007/978-3-319-24574-4_28.

[30] Kabir MM, Ohi AQ, Mridha MF. A multi-plant disease diagnosis method using convolutional neural network. ArXiv 2020. https://doi.org/10.1007/978-981-33-6424-0_7.

[31] Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC. MobileNetV2: Inverted Residuals and Linear Bottlenecks. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 2018. https://doi.org/10.1109/CVPR.2018.00474.

[32] Chollet F. Xception: Deep learning with depthwise separable convolutions. Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017- January, 2017. https://doi.org/10.1109/CVPR.2017.195.

[33] Simonyan K, Vedaldi A, Zisserman A. Deep inside convolutional networks: Visualising image classification models and saliency maps. 2nd Int. Conf. Learn. Represent. ICLR 2014 - Work. Track Proc., 2014.
[34] Pardede HF, Suryawati E, Zilvan V, Ramdan A, Kusumo RBS, Heryana A, et al. Plant diseases detection with low resolution data using nested skip connections. J Big Data 2020;7. https://doi.org/10.1186/s40537-020-00332-7.
[35] Nachtigall LG, Araujo RM, Nachtigall GR. Classification of apple tree disorders using convolutional neural networks. Proc. - 2016 IEEE 28th Int. Conf. Tools with Artif. Intell. ICTAI 2016, 2017. https://doi.org/10.1109/ICTAI.2016.75.
[36] Xie X, Ma Y, Liu B, He J, Li S, Wang H. A Deep-Learning-Based Real-Time Detector for Grape Leaf Diseases Using Improved Convolutional Neural Networks. Front Plant Sci 2020;11. https://doi.org/10.3389/fpls.2020.00751.