Data Augmentation Techniques on Chilly Plants to Classify Healthy and Bacterial Blight Disease Leaves

Sudeepthi Govathoti[1]
Department of Computer Science and Engineering
SoT, Gitam University, Hyderabad Campus-502329
Hyderabad, India

Deepthi Kamidi[3]
Department of Computer Science and Engineering
Vignan Institute of Technology and Sciences
Hyderabad, India

A Mallikarjuna Reddy[2], G BalaKrishna[4]
Department of Computer Science and Engineering
Anurag University
Hyderabad, India

Sri Silpa Padmanabhuni[5], Pradeepini Gera[6]
Department of Computer Science and Engineering
Koneru Lakshmaiah Education Foundation, Vaddeswaram
AP, India

***Abstract*—Designing an automation system for the agriculture sector is difficult using machine learning approach. So many researchers proposed deep learning system which requires huge amount of data for training the system. The proposed system suggests that geometric transformations on the original dataset help the system to generate more images that can replicate the physical circumstances. This process is known as "Image Augmentation". This enhancement of data helps the system to produce more accurate systems in terms of all metrics. In olden days when researchers work with machine learning techniques they used to implement traditional approaches which are a time consuming and expensive process. In deep learning, most of the operations are automatically taken care by the system. So, the proposed system applies neural style and to classify the images it uses the concept of transfer learning. The system utilizes the images available in the open source repository known as "Kaggle", this majorly consists of images related to chilly, tomato and potato. But this system majorly focuses on chilly plants because it is most productive plant in the South Indian regions. Image augmentation creates new images in different scenarios using the existing images and by applying popular deep learning techniques. The model has chosen ResNet-50, which is a pre-trained model for transfer learning. The advantage of using pre-trained model lies in not to develop the model from scratch. This pre-trained model gives more accuracy with less number of epochs. The model has achieved an accuracy of "100%".**

Keywords*—Image augmentation; geometric transformations; transfer learning; neural style learning; residual network

I. Introduction

The main goal of the data augmentation technique is to develop the model which is capable of handling the input that is unknown to the system and to develop a generalized training model. The proposed system focused on chilly plants but there is a situation where it can get similar leaf images of chilly but it is really chilly. So the proposed system uses the neural style learning to combine images of different plants and produce a new synthetic image. The image augmentation acts as a pre-processing step in the deep learning area to train the model. The proposed paper discusses the basic image manipulation operations, deep learning[5] and meta-learning techniques that can be performed on the image. The image augmentation operation is classified into two ways as shown in Fig. 1.

Offline mode stores the different augmented images in the hard disk, which requires high configuration of processor and RAM to run the program. Online mode utilizes the cloud services to store the image and it uses GPU's to run the program [19]. Google CoLab is an open source tool that run huge amount of images at a faster rate. Among the two types augmentations namely offline and online modes, this paper considers the online augmentation techniques since all the operations are performed in cloud using GPUs due to which waste of disk space is reduced. The offline data augmentation is preferred for the smaller datasets; the newly created images will be stored in the disk. This offline process is time-consuming and is expensive whereas in online or real data augmentation the transformation images occur randomly on different batches and the model is trained with more cases in each epoch in this method [8]. There three ways to generate augmented images as shown in Fig. 2.

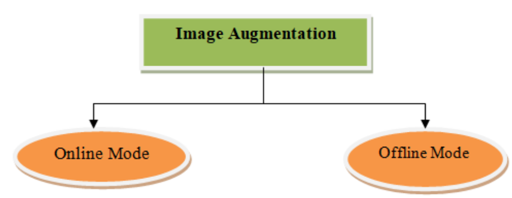


Fig. 1. Modes of Image Augmentation.

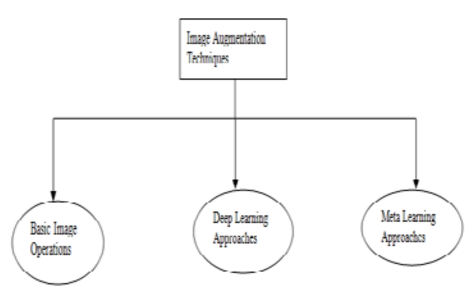


Fig. 2. Different Ways of Image Augmentation Techniques.

1) Basic image operations: The basic image operations include geometric transformations, which is a sequence of steps consisting of rotations, flip, resize, shift, zoom, crop, and adding of noise operations [9]. It also includes color space transformations; in general, images are composed of RGB color convention but for efficient processing of the data the image has to be transformed into gray-scale or other color saturation values. The basic operation also includes kernel filter, using convolution neural networks, the region of interest is extracted from the image. The image is stored as a two-dimensional matrix and the filter performs a dot product between the input and filter layers and the values are added to get a single value in each position. Kernel filtering helps in edge detection, image sharpening, and blurring operations [10]. The latest improvement has included random erasing techniques to erase the pixels in the image by selecting a rectangular region, an area is selected based on the probability which is calculated with help of aspect ratio and area ratio. It plays a vital role in image classification, object detection, and person re-identification. It can be easily integrated with the neural networks and it can use on pre-trained models. The basic image operations categories are presented in Fig. 3.

2) Deep learning approaches: These approaches consist of adversarial training, this type of approach is highly needed in the models where Gaussian noise is injected and images are transformed with worst-case perturbations, this results in incorrect answer with high confidence values [11]. Deep learning also involves neural style transferring technique which helps to create new images by blending images especially when the model has content or style reference images. It uses two different distance functions, one is used to identify the differences in terms of content, and the other is used to identify the differences in terms of style. The third and important approach is the usage of GAN models, an unsupervised technique that can generate new images by learning from the patterns that exist in between the input image. GAN's consists of two components: generator to create new images by training the model and discriminator for classification purpose [12, 13]. In GAN's, the generator takes input as a fixed-length vector, which is known as "Noise Vector" for producing the salt and pepper noise images because most of the plants have smoked layer above them. Generator produces outputs samples that belong to the domain of leaves but with different styles. Discriminator takes the input from the domain and outputs a binary value based on the prediction performed. The task of this component is predict whether the scanned image is a real or augmented image. Higher the misclassification produced by this component higher will be accuracy of system because more number of augmented images has passed the test. So in designing the GAN, the major focus should be on layers of the generator. The general architecture of GAN is presented in the Fig. 4.

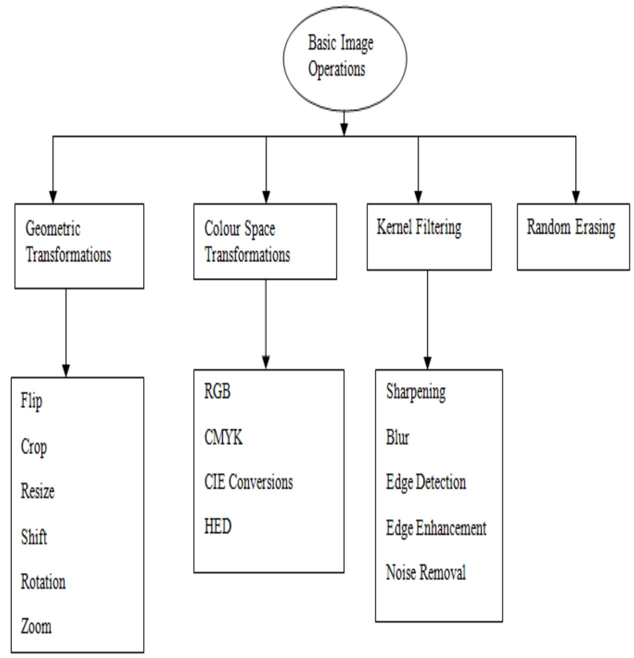


Fig. 3. Summary of Basic Image Operations.

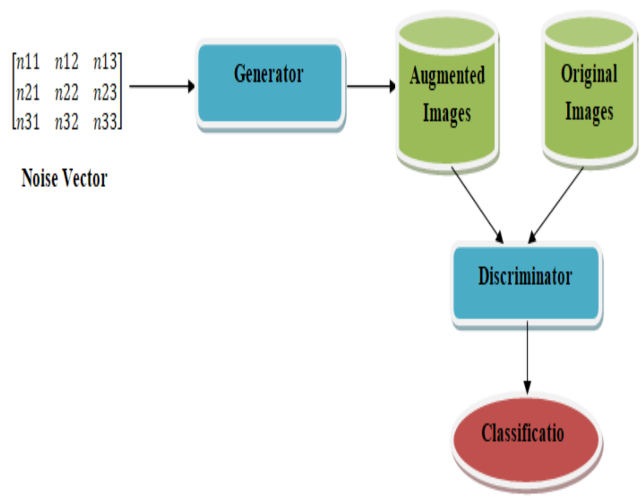


Fig. 4. GAN Architecture.

3) Meta-Learning approaches: These approaches are used to balance the data between the real world and the simulated world. Recently Google has designed a novel method known as "AutoAugment", it is a policy-based approach to identify the best augmentation suited for the dataset. The search space of the AutoAugment has 5 sub-policies with 2 operations each. This process of AutoAugment will save us a lot of time in checking and applying all the possible combinations of operations that can be performed to increase our dataset size. The selection of operation to be performed is based on the dataset system supply, this automatic selection of operation is acquired from the reinforcement learning. Fig. 5 illustrates the process of Auto Augmentation using the optimizers.

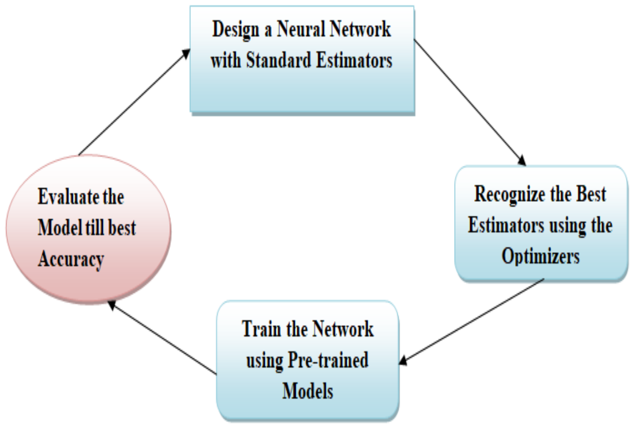


Fig. 5. Auto Augmentation Process.

The design considerations for augmentation involve train-time data augmentation to reduce the generalization error and test-time data augmentation to improve predictive performance. The most important operation that should be performed is resizing operation that all the images should be of the same size and some pixel scaling and normalization operations should be performed.

The paper first presented introduction section, which describes about different augmented images generation techniques in online mode. The second section literature analyzes the previous works to identify the gaps in the research. The proposed model section discusses about the basic image operations along with the ResNet-50 CNN for classification [15]. The results section discusses about both proposed results along with the compared results of previous works in terms of every metric. The last section i.e., conclusion presents the advantages of pre-trained model then it extends the section by coining the limitations of the present work.

II. Literature Review

Many Research Scholars and Scientists are continuously working on various plant diseases to improve the quality of production using machine learning and deep learning techniques. All these researchers have considered the controlled conditions and obtained the plant images from PlantVillage dataset. The application of computer vision and deep learning techniques has given good accuracy systems for automatic detection of diseases in plants.

Quan Huu Cap[1] developed a Generative Adversarial Networks known as "LeafGAN", which act as a data augmentation tool by creating new diseased images from the healthy images. This system preserves the background of the image as well as generates the images of high quality. The core component of this model is LFLSeg, which is a label-free and weakly supervised segmentation module. In this model, it uses feature maps to extract the segmentation information[6], and this in return helps the model to learn about dense and interior regions of the leaf images implicitly. The output of the segmented image is projected as a heat map so that it calculates the probability of each pixel in the final decision. Finally, this model has boosted the diagnosis system and it solved the problems that arise due to the overfitting of the data.

Haseeb Nazki[2] proposed Pipelined Generative Adversarial Networks for plant disease detection to address the problem of imbalanced data shifting. The working of this model consists of two major components. The first component is AR-GAN, which is used for generating the data augmentation synthetically by translating an image from one domain into another domain. A parameter known as discriminator decides whether an image belongs to a particular domain or not. Based on the dataflows and loss functions, the image is reconstructed so that its performance is improved over cycle GAN. AR-GAN has a network that has an activation reconstruction for feature extraction. The second component is RESNET-50 CNN for disease detection with the same configuration as the baseline with RELU as activation functions [16]. The network is fine-tuned by using ImageNet pre-trained weights.

Daniel Ho[3] suggested a population-based augmentation algorithm, which uses a dynamic strategy. The main goal of this algorithm is to optimize the hyperparameter by using schedule learning policies. The PBA first runs a gradient descent algorithm on each epoch and then it does the evaluation on the validation data. Truncation selection is applied to the bottom 25% of the data based on the weights and top25% of the data based on the hyperparameters.

Sungbin Lim [4] proposed the Fast AutoAugment Algorithm based on density matching. The model constructs a search space for the images and defines two efficient operations known as calling probability and the magnitude. The search strategy uses the probability distribution on a pair of training datasets, the values are parameterized and evaluated the accuracy and loss values. The performance is measured by comparing the amount of the data that is similar in both datasets and it uses K-fold stratified shuffling. At last, the policy is explored by using Bayesian Optimization by employing a kernel density estimator. Table I illustrates the advantages and disadvantages of the previous works.

TABLE I. COMPARATIVE ANALYSIS

S.No	Author Name	Algorithm Name	Merits	Demerits
1	Quan	LeafGAN	The classification process by taking the segmented images is easy for the GAN	The model has used traditional segmentation approach for feature extraction. The annotation process is difficult
2	Haseeb	ARGAN	The residual component used tries to minimize the loss function at every iteration	The usage of ImageNet which less number of leaf images for knowing the weights of the layer has reduced the accuracy of the model.
3	Daniel	PBA	During the process of distribution of data the model has wisely chosen mixture of top and bottom layers of data for validation purpose	The learning policy implemented by the population is difficult since all the combinations of learning policies are to be computed
4	Sungbin	Fast Auto Augment	The model uses the probability to construct the search space. This helps the model to have very small search area to recognize the desired regions	The density parameters are found using the optimization technique based on bayes theorem. But since it is applied for K-fold, it is very expensive process

III. PROPOSED MODEL

The aim of the proposed model is to develop basic online augmentation techniques along with convolutional neural networks[7] and calculate the accuracy for the identification of bacterial blight disease in chilly plants.

A. Basic Image Augmentation Operations

The main advantage of these geometric operations is that even though various operations are performed on these images, the features of the image remain the same. All these operations are performed with the help of ImageDataGenerator class of the Tensorflow library and all the images captured will have different dimensions so to maintain all the images with the same dimensions the common operation of resizing is applied to all the images [18].

1) Rotation: The basic operation that can be applied to any image to change the orientation of the image is rotated. The rotation angle is specified in terms of degrees which can accept the values from 0 to 360 in the clockwise direction. [14].

2) Flipping: It is an extension of rotation operation. Flipping operation is used to perform the transposition of a row and column pixels. Here, two types of flipping operations can be performed. Depending on the nature of the images, either of the horizontal or vertical flips can be applied.

3) Shearing: The process in which the pixels can be shifted from one position to another position either horizontally or vertically is known as “Shearing”. Here, the dimensions of the mage remain the same i.e., few pixels will be clipped off.

4) Cropping: In general, the images contain a region of interest at various locations. To find that location the best operation that one can use is cropping the image at the required place. Since the proposed system is a classification model, the output generated by the model should be the subset of the whole image.

5) Zooming: Zoom operation either adds new pixel values or interpolates pixel values. Let us consider the value specified is x then zoom performs 1+x and 1-x operations around the pixels of the image. It helps in the uniform sampling of the image.

6) Brightness or contrast: This operation is used to perform either to darken or brighten the image. This operation mainly helps the model to train the system under various lighting conditions. Here, we can specify the minimum and maximum value as a percentage. A value of less than 1.0 darkens the image and a value greater than 1.0 brightens the image. All the sub sections of Fig. 6 show the output of different image operations on the chilly leaves. The order of selection for these operations doesn’t have any impact on the augmentation and classification process [20]. There are many image manipulation operations but the proposed system implements only five of the important operations so that the burden on the neural network gets reduced.

B. Basic Geometric Image Operations Algorithm

Step 1: All the images should be of the same size. To do this, we apply to rescale the images to 1/255, because the minimax pixel value is 255.

Step 2: Image is rotated to 40^0 to make the image out of the frame and used nearest neighbor fill interpolation.

Step 3: Apply both horizontal shift and vertical shift with 20%.

Step 4: Apply the sheer range and zoom range of 20%.

Step 5: Apply horizontal flip.

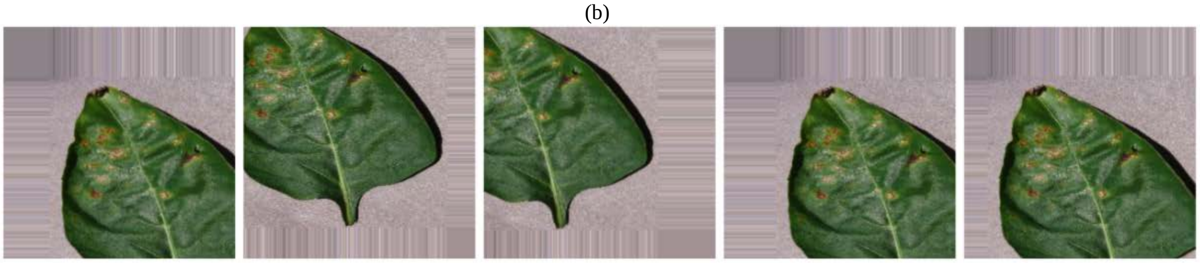
The algorithm will give the augmented images as follows. A sample screenshot is shown in Fig. 7.

C. Applying Convolution Neural Networks

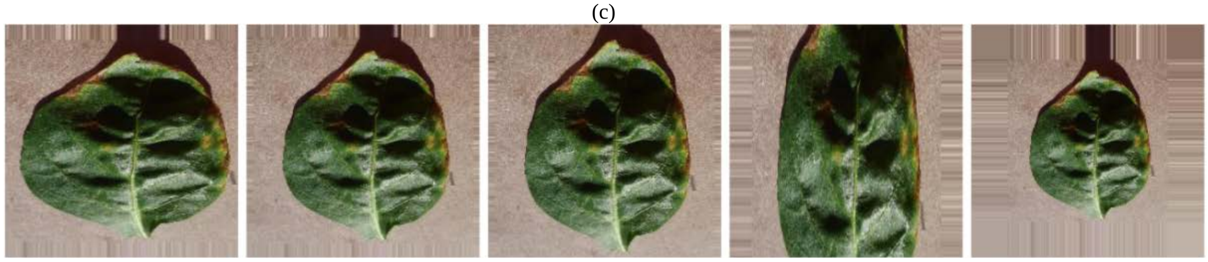
2D convolution takes an image as input and passes it through a kernel filter to calculate the dot product. The weighted matrix is converted into a feature matrix. These features are important to find the nearest weights to identify the region of interest. This helps to reduce the number of pixel operations to perform. In the proposed paper, the model uses padding with value as the same, which determines the amount of the pixels to be added to the image and add layers with pixel value as zeros. In general, the pixels stored in the center are used more than corners and edges. So to preserve the edge's information, padding helps a lot. The same value for the attribute padding is illustrated as follows:

(a)

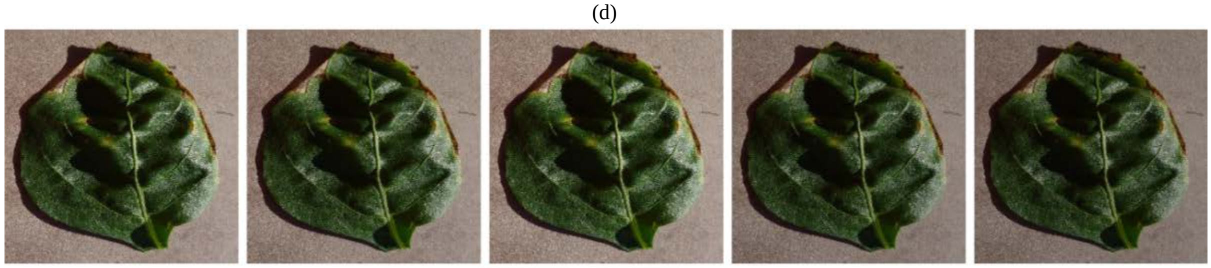
(b)



(c)



(d)



(e)

Fig. 6. (a) Output of Images Generated Randomly at the Angle of 95 Degrees, (b) Output Images Generated by Performing Horizontal Flip, (c) Output Images Generated by Performing Shearing Operation with Both width and Height, (d) Output Images Generated by Performing Zoom, (e) Output Images Generated by Performing Brightness.

Fig. 7. Screenshot of Augment Images Created using the Online Data Augmentation Process.

Let, the number of layers to be added be as the border of an image be (wXw) and the image size be (mXm). After padding the image, the size of the image becomes as (m+2w). (m+2w). Let the kernel filter size be (yXy), then after applying the kernel filter, the output image size is (m+2w-y+1). (m+2w-y+1). The padding maintains the same size as of input image by using the notation shown in equation (1).

$$length\_padding = \frac{[(m+2*w)*(m+2*w)]*y^2}{m*m} \text{ -} \quad (1)$$

The transfer function or activation function determines the output of the node in the neural network. This converts the corresponding output into binary values. The proposed paper uses ReLU, a nonlinear transfer function, which gives the output as the maximum value of the input. It is a widely used transfer function because it implements backpropagation and as well as it doesn't activate all the values simultaneously [17]. The equation of the function is shown in (2).

$$f(input\_feature) = 0, if\ input\_feature\ < 0$$

= 1, if input_feature >= 0 – (2)

The proposed system uses the CIFAR-100 predefined dataset with 100 class labels using the ResNet-50 to get the proper weights for training the network. After training the model, the augmentation techniques can create different feature maps and sometimes lower resolution pixels may contain important structural elements. So, these issues can be solved by designing pooling layers. The pooling layers create a subset of feature maps for each operation separately. It detects the features irrespective of augmentation and noises contained in the image. In the proposed paper, the model used the max-pooling layer. It takes the maximum value that occurs in the filter applied. The main advantage of pooling in smaller datasets is it can avoid the overfitting problem by implementing the dropout regularization mechanism. For having a good performance, the dropout value for the hidden layers can be between 0.5 and 0.8.

Now the pooled feature map should be transferred into the column, because the neural network can accept only the long vector of inputs. Then, a dense layer is applied to the neural network, which calculates the dot product of the input image and weighted data i.e., kernel, and added to the bias value with the usage of activation function to optimize the model. Finally, the model is compiled with the help of Adam optimizer, it updates the network weights iteratively based on the training data. It is popular for its fastness by updating velocity and momentum parameters as shown in equations (3) & (4).

$$Momentum_i = \beta * Momentum_{i-1} + (1-\beta) * gradientvalue \text{ -} \quad (3)$$

$$Velcoity_t = \alpha * velocity_{t-1} + (1-\alpha) * momentum_i \text{ -} \quad (4)$$

The extracted features from the chilly plants after training the model are presented in Fig. 8. This figure shows the layer by layer output of CNN for better visualization of essential features.

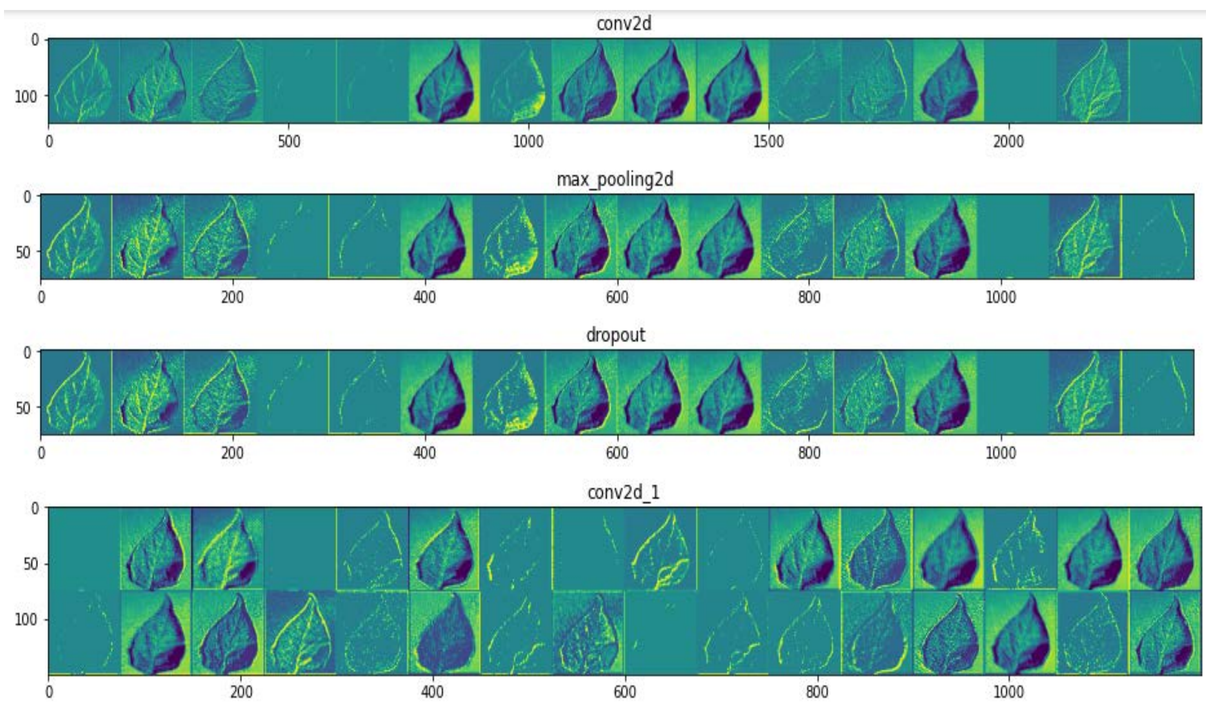


Fig. 8. Screenshot of Feature Maps after Every Layer in CNN.

IV. Experimental Results

Since the model uses the pre-trained model ResNet-50, the number of trainable parameters get reduced from 1,67,300 to 10,400. So the model can claim that it has good dimensionality reduction. The system to overcome the drawbacks of the machine learning models it stacks layers of network to learn the complex features with efficient learning rate. The proposed system uses that stacked layer architecture which consists of a pile of conv2d, activation and batch normalization layer. The major advantage of this network lies in reducing the dimensions drastically at every step of training. The model also uses the best estimators like using Adam optimizer, learning rate as 0.5 and others. The systems presents a sample summary of the training model in the Fig. 9 to understand the underlying parameters.

```
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 32, 32, 3)]  0
__________________________________________________________________________________________________
conv2d (Conv2D)                 (None, 32, 32, 16)   448         input_1[0][0]
__________________________________________________________________________________________________
batch_normalization (BatchNorma (None, 32, 32, 16)   64          conv2d[0][0]
__________________________________________________________________________________________________
activation (Activation)         (None, 32, 32, 16)   0           batch_normalization[0][0]
__________________________________________________________________________________________________
conv2d_1 (Conv2D)               (None, 32, 32, 16)   272         activation[0][0]
__________________________________________________________________________________________________
batch_normalization_1 (BatchNor (None, 32, 32, 16)   64          conv2d_1[0][0]
__________________________________________________________________________________________________
activation_1 (Activation)       (None, 32, 32, 16)   0           batch_normalization_1[0][0]
__________________________________________________________________________________________________
conv2d_2 (Conv2D)               (None, 32, 32, 16)   2320        activation_1[0][0]
__________________________________________________________________________________________________
batch_normalization_2 (BatchNor (None, 32, 32, 16)   64          conv2d_2[0][0]
__________________________________________________________________________________________________
activation_2 (Activation)       (None, 32, 32, 16)   0           batch_normalization_2[0][0]
__________________________________________________________________________________________________
```

Fig. 9. First 2 Layers Summary in the ResNet-50.

```
Epoch 1/15
49/49 [==============================] - 36s 733ms/step - loss: 0.2839 - accuracy: 0.9122 - val_loss: 0.7233 - val_accuracy: 0.4000
Epoch 2/15
49/49 [==============================] - 26s 527ms/step - loss: 0.1507 - accuracy: 0.9183 - val_loss: 0.9869 - val_accuracy: 0.3900
Epoch 3/15
49/49 [==============================] - 25s 502ms/step - loss: 0.1114 - accuracy: 0.9524 - val_loss: 0.3323 - val_accuracy: 0.8400
Epoch 4/15
49/49 [==============================] - 25s 502ms/step - loss: 0.0494 - accuracy: 0.9835 - val_loss: 0.3017 - val_accuracy: 0.8500
Epoch 5/15
49/49 [==============================] - 25s 502ms/step - loss: 0.0485 - accuracy: 0.9886 - val_loss: 0.2070 - val_accuracy: 0.9200
Epoch 6/15
49/49 [==============================] - 24s 499ms/step - loss: 0.1324 - accuracy: 0.9462 - val_loss: 0.7213 - val_accuracy: 0.5300
Epoch 7/15
49/49 [==============================] - 24s 498ms/step - loss: 0.0929 - accuracy: 0.9659 - val_loss: 0.3181 - val_accuracy: 0.7900
Epoch 8/15
49/49 [==============================] - 25s 500ms/step - loss: 0.0611 - accuracy: 0.9741 - val_loss: 0.4429 - val_accuracy: 0.7900
Epoch 9/15
49/49 [==============================] - 25s 502ms/step - loss: 0.0557 - accuracy: 0.9659 - val_loss: 0.6358 - val_accuracy: 0.5800
Epoch 10/15
49/49 [==============================] - 25s 502ms/step - loss: 0.0165 - accuracy: 0.9928 - val_loss: 0.1411 - val_accuracy: 0.9200
Epoch 11/15
49/49 [==============================] - 25s 502ms/step - loss: 0.0040 - accuracy: 0.9990 - val_loss: 0.0612 - val_accuracy: 0.9700
Epoch 12/15
49/49 [==============================] - 25s 503ms/step - loss: 0.0061 - accuracy: 0.9990 - val_loss: 0.2557 - val_accuracy: 0.9100
Epoch 13/15
49/49 [==============================] - 25s 503ms/step - loss: 7.3316e-04 - accuracy: 1.0000 - val_loss: 0.3898 - val_accuracy: 0.9000
Epoch 14/15
49/49 [==============================] - 25s 507ms/step - loss: 2.6175e-04 - accuracy: 1.0000 - val_loss: 0.2725 - val_accuracy: 0.9100
```

Fig. 10. Evaluation Metrics in all Epochs.

Fig. 10 shows the accuracy for every iteration by setting the epoch value to 15. It computes accuracy and loss of training data and it also computes the accuracy and loss of validation data. From the figure, it is evident that the model has started with good accuracy of “91.22” and slowly it has reached to 100% accuracy.

The evaluation of deep learning algorithms is measured using the both training and testing data. Fig. 11 visualizes the metrics obtained in the Fig. 10 as a graph for the users to understand the nature of the system. Fortunately, in case of training data, the model performed stable by getting more accuracy for training than testing data and more over it has increased gradually. In the loss graph, it is clearly evident that the loss is also decreasing gradually. The loss rate is also very less.

The below section compares the output of the proposed model with the previous works studied in the literature section to prove the efficiency of the model. Fig. 12 visualizes the graphs obatined by the algorithms in terms of different metrics. X-axis repesents methodology and Y-axis represents the measuring scale.

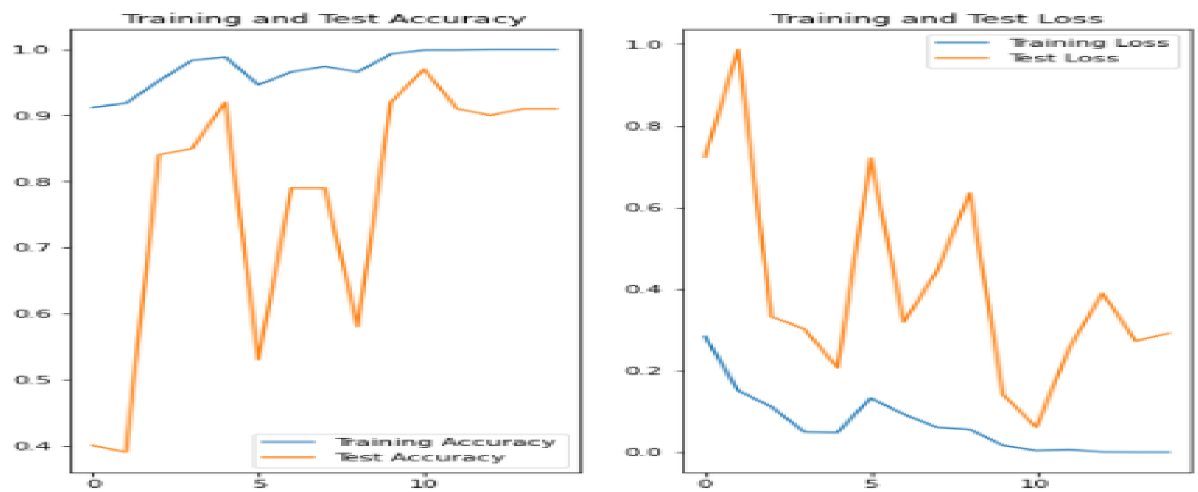


Fig. 11. Screenshot of Training, Test Accuracy.

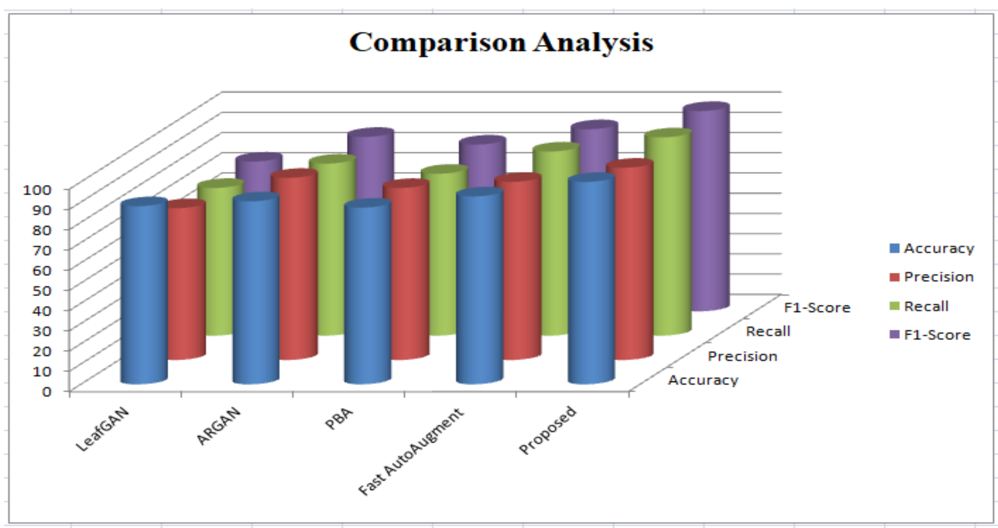


Fig. 12. Comparison Analysis over Metrics.

V. Conclusion

In a real-time environment the conditions are uncontrolled and the system may face problems like labeling of disease dataset, and addressing the data imbalance problems. The size of dataset is smaller in size in the real-time environment so they cannot work properly. The traditional systems take a lot of computations to select and apply all the possible geometric transformations. So, the proposed system by designing a good policy schema like using pre-trained model to gets the weights of neural networks and applying five basic transformations to create augmented images helps the model to design an accurate system. In future, the model can be transformed to use GAN approaches to create images because the model has achieved 100% accuracy whose state is claimed as “Overfitting”.

References

[1] Saikawa, Takumi & Cap, Quan & Kagiwada, Satoshi & Uga, Hiroyuki & Iyatomi, Hitoshi. (2019). AOP: An Anti-overfitting Pretreatment for

Practical Image-based Plant Diagnosis. 5177-5182. 10.1109/BigData47090.2019.9006567.

[2] Nazki, Haseeb. (2018). Synthetic Data Augmentation for Plant Disease Image Generation using GAN.

[3] Ho, Daniel & Liang, Eric & Stoica, Ion & Abbeel, Pieter & Chen, Xi. (2019). Population Based Augmentation: Efficient Learning of Augmentation Policy Schedules.

[4] Lim, Sungbin & Kim, Ildoo & Kim, Taesup & Kim, Chiheon & Kim, Sungwoong. (2019). Fast AutoAugment.

[5] Ayaluri MR, K. SR, Konda SR, Chidirala SR. 2021. Efficient steganalysis using convolutional auto encoder network to ensure original image quality. PeerJ Computer Science 7:e356 https://doi.org/10.7717/peerj-cs.356.

[6] BANGARE, S.L., PRADEEPINI, G. and PATIL, S.T., 2018. Regenerative pixel mode and tumour locus algorithm development for brain tumour analysis: A new computational technique for precise medical imaging. *International Journal of Biomedical Engineering and Technology*, **27**(1-2), pp. 76-85.

[7] Grandhe, Padmaja. (2020). A Novel Method for Content based 3D Medical Image Retrieval System Using Dual Tree M-Band Wavelets Transform and Multiclass Support Vector Machine. Journal of Advanced Research in Dynamical and Control Systems. 12. 279-286. 10.5373/JARDCS/V12I3/20201192.

[8] Ilaiah Kavati, A. Mallikarjuna Reddy, E. Suresh Babu, K. Sudheer Reddy, Ramalinga Swamy Cheruku,Design of a fingerprint template protection scheme using elliptical structures,ICT Express,Volume 7, Issue 4,2021,Pages 497-500,ISSN 2405-9595,https://doi.org/10.1016/j.icte.2021.04.001.

[9] Sahu, A.K., & Swain, G. (2018). A Novel n-Rightmost Bit Replacement Image Steganography Technique. *3D Research, 10*, 1-18.

[10] Padmanabhuni, Ms. S. S. (2020). An Extensive Study on Classification-based Plant Disease Detection Systems. Journal of Mechanics of Continua and Mathematical Sciences, 15(5). https://doi.org/10.26782/jmcms.2020.05.00002.

[11] A Mallikarjuna Reddy, Vakulabharanam Venkata Krishna, Lingamgunta Sumalatha and Avuku Obulesh, "Age Classification Using Motif and Statistical Features Derived On Gradient Facial Images", Recent Advances in Computer Science and Communications (2020) 13: 965. https://doi.org/10.2174/2213275912666190417151247.

[12] KUMAR, P., PRADEEPINI, G. and KAMAKSHI, P., 2019. Feature selection effects on gradient descent logistic regression for medical data classification. International Journal of Intelligent Engineering and Systems, 12(5), pp. 278-286.

[13] Divya, T. V., & Banik, B. G. (2021). Detecting Fake News Over Job Posts via Bi-Directional Long Short-Term Memory (BIDLSTM). In International Journal of Web-Based Learning and Teaching Technologies (Vol. 16, Issue 6, pp. 1–18). IGI Global. https://doi.org/10.4018/ijwltt.287096.

[14] Kishore, P.V.V., Kumar, K.V.V., Kiran Kumar, E., Sastry, A.S.C.S., Teja Kiran, M., Anil Kumar, D. & Prasad, M.V.D. 2018, "Indian Classical Dance Action Identification and Classification with Convolutional Neural Networks", Advances in Multimedia, vol. 2018.

[15] C. R. T, G. Sirisha and A. M. Reddy, "Smart Healthcare Analysis and Therapy for Voice Disorder using Cloud and Edge Computing," *2018 4th International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, Mangalore, India, 2018, pp. 103-106, doi: 10.1109/iCATccT44854.2018.9001280.

[16] Soumya Ranjan Nayak, Jibitesh Mishra, G. Palai, A modified approach to estimate fractal dimension of gray scale images, Optik, Volume 161, 2018, Pages 136-145, ISSN 0030-4026, https://doi.org/10.1016/j.ijleo.2018.02.024.

[17] A. M. Reddy, K. SubbaReddy and V. V. Krishna, "Classification of child and adulthood using GLCM based on diagonal LBP," 2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Davangere, 2015, pp. 857-861, doi: 10.1109/ICATCCT.2015.7457003.

[18] Akbar, S., & Midhunchakkaravarthy, D. (2020). A Novel Filtered Segmentation-Based Bayesian Deep Neural Network Framework on Large Diabetic Retinopathy Databases. In Revue d'Intelligence Artificielle (Vol. 34, Issue 6, pp. 683–692). International Information and Engineering Technology Association. https://doi.org/10.18280/ria.340602.

[19] Swarajya Lakshmi V Papineni, Snigdha Yarlagadda, Harita Akkineni, A. Mallikarjuna Reddy. Big Data Analytics Applying the Fusion Approach of Multicriteria Decision Making with Deep Learning Algorithms *International Journal of Engineering Trends and Technology*, 69(1), 24-28, doi: 10.14445/22315381/IJETT-V69I1P204.

[20] A. M. Reddy, V. V. Krishna, L. Sumalatha and S. K. Niranjan, "Facial recognition based on straight angle fuzzy texture unit matrix," 2017 International Conference on Big Data Analytics and Computational Intelligence (ICBDAC), Chirala, 2017, pp. 366-372, doi: 10.1109/ICBDACI.2017.8070865.