IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

A Deep Learning-Based Recognition Technique for Plant Leaf Classification

PAUL SHEKONYA KANDA[1], KEWEN XIA[1], AND OLANREWAJU HAZZAN SANUSI[2]

[1]School of Electronic and Information Engineering, Hebei University of Technology, No. 5340 XiPing Road, Beichen District, 300401, Tianjin, P. R. China.
[2]College of Intelligence and Computing, No. 135 Yaguan Road, Haihe Education Park, Jinnan District, 300350, Tianjin, P.R. China.

Corresponding author: Prof. Xia Kewen (e-mail: kwxia@hebut.edu.cn).

ABSTRACT In the practice of plant classification, the design of hand-crafted features is more dependent on the ability of computer vision experts to encode morphological characters that are predefined by botanists. However, the distinct features that each plant has as demonstrated by its leaves can be automatically learned based on the end-to-end advantage of Deep Learning algorithms. Therefore, Deep Learning based plant leaf recognition methods is an important approach nowadays. In this article, we are applying three technologies to achieve a model with high accuracy for plant classification. A Conditional Generative Adversarial Network was used to generate synthetic data, a Convolutional Neural Network was used for feature extraction and the rich extracted features were fed into a Logistic Regression classifier for efficient classification of the plant species. The effectiveness of this method can be seen in the wealth of plant datasets that it was tested on. The paper contains results on seven datasets with different modalities. We utilized both Deep Learning and Logistic regression in effectively classifying the plants using their leaf images with accuracies averaging 96.1% for about eight datasets used, but greater for the individual datasets from 99.0 to 100% on some individual datasets. Extensive experiments on each of the datasets demonstrate the superiority of our method compared with others and are highlighted in our results.

INDEX TERMS cGAN, classification, convolutional neural networks, deep learning, feature extraction, leaf images,

I. INTRODUCTION

ACCORDING to the State of Plants report [1], there exist about 390,000 plant species known to modern science. This makes it difficult for a botanist or an expert to be able to identify and classify all these species. Even more so impossible for non-experts. Not forgetting the fact that some plant species have high similarities and the differences being so little pose the need for a fine-grained classification approach. The leaves of plants contain a considerable amount of information regarding the particular plant species and a lot can be learned about a plant just from its leaves. Plants, obviously, right from the inception of creation have had a variety of use for humans and are not stopping any moment soon, ranging from their use for food, medicine, shelter, etc.

In addition, owing to global warming and other factors, many plants nowadays are faced with the challenge of extinction. Non-endangered as well as endangered plant species need to be preserved and conserved adequately to counter this risk. Hence, there is a need to develop an automated or computerized system to identify and classify plants in an efficient manner. Therefore, the need for adequate knowledge and management of plant species cannot be overemphasized. The traditional artificial identification methods are based on plant morphology, which though having its form of success is subjective, costlier in terms of manpower and proves inefficient in the long run. Fig. 1 shows the general leaf features used in the classification of plant species. Plant leaves are two-dimensional in nature; therefore, making it possible to engage in the identification of plant leaves automatically using image-processing techniques. In recent times, Deep Learning and in particular the use of Convolutional Neural Networks (CNNs) have proven well suited for addressing computer vision problems of which plant classification can be considered to be one. Deep learning eliminates the need for domain expertise and hard-core feature extraction that only expert botanists can provide.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

instead, a series of consecutive convolution operations between the input image and convolutional filters, form the feature map and extract discriminative patterns from individual plant leaves. Several image recognition methods and applications based on deep learning are easily accessible, and they are still attracting a lot of attention. Some of these methods could be applied to the field of botany and agriculture to tackle problems computationally.

Table 1 below shows the general leaf features that are widely studied in classification. Features such as shape, color, and texture can be seen to be the generic distinguishing features that separate plant leaves and other details sub-features can be grouped within the three as shown in Table 1. This paper proposes a method based on an ensemble of some Deep Learning techniques for efficient Plant Classification applied on plant leaf images.

The main contributions of this paper are summarized as follows:

- Utilizing a blending of three technologies to improve the accuracies of plant species classification via plant leaf images. This is novel in the area of botany and plant classification and it reliably produces state-of-the-art results. The results can be compared with other previous results in this field. Tables 13-15 display such results.
- The utilization of a conditional Generative Adversarial Network to tackle the problem of a lack of sufficient training data or uneven class balance that could be found within datasets in performing Deep Learning tasks. This serves to augment leaf image datasets, which have not been large enough, as this field still lacks a large number of datasets for adequate training of Deep Neural Networks for better generalization.
- The result of the augmented leaf dataset produced more than 3.0% increase in accuracy. This is a milestone in the field of Deep Learning.

The rest of the paper is organized as follows: Section II reviews previous literature with related work done on plant classification, Section III details the various datasets we used, describes our approach and our experiment parameters, while Section IV presents the experiments carried out with parameters used; Section V gives the discussion of results, and also provides a comparison with conventional solutions. Finally, conclusions are drawn in Section VI.

II. RELATED WORK

There are lots of methods that are used for classification in Machine Learning. To train the computer to adequately and correctly classify objects is a huge task that artificial intelligent practitioners have been working hard on for many years now and plant classification is no different in this feat.

Variations on leaf characteristics are preferably employed in automated plant identification systems using computer vision methods because of leaves being easily observable, accessible, and describable compared to other plant organs. Kadir et al. [2], as well as Cope et al. [3], and Ahmed et al. [4], give comprehensive surveys on methods for automated plant identification. However, plant identification is still considered to be a challenging and unsolved problem since all classical computer vision employs handcrafted methods that are dependent on chosen natural features given the extreme diversity of botanical data. For example, Jin et al. [5], employed a classical image processing chain of image binarization to separate the background and the leaf from a leaf image, detection of contours and contour corners, and geometrical derivations of special leaf tooth features. This approach was then evaluated on eight different species of plants and Jin et al. [5] reported in-species specific identification accuracies ranging between 72.8% and 79.3%. But obviously, this approach solely cannot deal with species showing any significant appearances of leaf teeth and the percentage accuracy is still far from the optimum.

Many hand-crafted features can be selected from plant leaf images, most of which fall into either shape, texture, or venation. Most of the previous studies on this depended on shape recognition techniques to model and represent the contour shape of the leaf due to the diversity of leaf shapes. Neto et al [6] used Elliptic Fourier and discriminant analyses to show the distinction in plant species based on their unique leaf shapes. [7] proposed two shape modeling approaches based on the invariant-moments and centroid-radii models. Du et al [8] also went on to propose a technique based on combining geometrical and invariant moments features to extract morphological structures of leaves.

Shape Context (SC) and Histogram of Oriented Gradients (HOG) have also been useful approaches in the attempt of creating an efficient leaf shape descriptor [9][10]. Recently, Aakif and Khan [11] proposed using different shape-based features such as morphological characters, Fourier descriptors, and a newly designed Shape-Defining Feature (SDF). Although the algorithm showed its effectiveness in baseline leaf datasets like Flavia [12], the SDF is highly dependent on the segmented result of leaf images and doesn't produce optimum performance. Hall et al. [13] proposed using Hand-Crafted Shape (HCS) and Histogram of Curvature over Scale (HoCS) [14] to analyze leaves. Larese et al. [15] recognized legume varieties based on leaf venation in which they first segmented the vein pattern using Hit or Miss Transform (UHMT), then going further to use LEAF

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

GUI measures to extract a set of features for veins and areoles but it was computationally expensive.

Wilf et al. [16] applied their approach to the vein features of leaves and report a classification accuracy of 72.14% for 19 leaf families. However, the images are taken from cleared specimens that are prepared laboriously and there is still a limitation to a restricted class of features. Kumar et al. [14], proposed a mobile app, called LeafSnap, to enable users to identify trees from leaf photographs. It achieves a top-1 recognition rate of about 73% for 184 tree species, however, a higher accuracy could still be achieved.

The last step towards model-free approaches to plant identification is to get rid of hand-crafted features. In the last years, deep learning convolutional neural networks (CNNs) have seen a significant breakthrough in computer vision, especially in the field of visual object categorization, Krizhevsky et al. [17], due to the rise of efficient general-purpose computing on graphics processing units (GPGPU provides high degrees of parallelization) and the availability of largescale image data (in publicly available datasets, on the internet, in social media, (specialized) social networks, etc.) that provide the data amount necessary for training deep CNNs with thousands of parameters.

An essential advantage of deep CNNs is the automatic learning of task-specific representations of the input data which replace traditional feature-based representations using hand-crafted features but there's still the challenge of obtaining optimum accuracy in classification using CNNs.

Lee et al. [18], presented a CNN approach to taxon identification based on leaf images and reported an average accuracy of 99.7% on a dataset covering 44 species. Zhang et al. [19], used CNN to classify the Flavia dataset and obtained an accuracy of 94.69%. Goëau et al. [20], report on the plant identification task PlantCLEF 2016 that was organized within the ImageCLEF initiative2 dedicated to the system-oriented evaluation of visual-based plant identification but there's still a need for higher accuracy in this classification.

Another study by [21] attempted deep learning in plant identification using vein morphological patterns. They first extracted the vein patterns using UHMT, and then trained a CNN to recognize them using a central patch of leaf images, yet, accuracy and complexity are still problems to combat, and having the model generalize well is still a great need. Therefore, in the follow-up study, we should focus on how to improve identification accuracy. Also, deep learning methods have some limitations. If the number of available images is less than the number of images required for state-of-the-art-based models such as GoogleNet [22], AlexNet [17], and so on, we will be challenged to train the network having only a handful of leaf images in our datasets. This is because CNNs on a small dataset surely get overfitting. To overcome this problem, the number of training data should be increased. But providing a large amount of training data is very difficult and costly. Hence, in this work, we employed the use of Conditional Generative Adversarial Networks to generate synthetic data to augment the datasets. Additionally, we used a state-of-the-art CNN model to perform feature extraction, it can extract rich features from images; and then applied a reliable and efficient classifier to classify the plants given the extracted features. Results show great performance on the various evaluation metric that we used.

III. PROPOSED METHOD

Deep learning, being a special class of machine learning algorithms is of high importance in the field of Intelligent Computing. Generative Adversarial Network(GAN) was first proposed by Goodfellow *et al.* [23] and they have had tremendous success in the field of Deep Learning. Similarly, a deep Convolutional Neural Network (CNN) can extract higher-level features progressively from the input images given to it by the multiple layers used in a model. The Residual Network [24] architecture has proved to be a reliable architecture over the past few years since its inception and coming 1st place on the ILSVRC 2015 classification task. It has lots of success in other research works with almost 30,000 reference citations to prove its great contribution and effectiveness in the field of Deep Learning, however, applying it alone does not give us the optimal classification accuracy we desire in plant classification via the leaf images, hence our ensemble with other DL methods. For our research, we utilized a Residual Network model, with 50 layers, which was pre-trained with the state-of-the-art ImageNet dataset. We applied it to extract the features of the leaf images in our respective datasets.

Figure 7 shows the flowchart of the overall framework of our model. It consisted mainly of four main steps as shown in the figure, which is, image pre-processing, data augmentation, feature extraction, and image classification. First, the leaf images were pre-processed and fed into the conditional Generative Adversarial Network, which serves as the means for extra data augmentation for the Deep Residual network to have more data. Afterward, the Deep Residual Network, which has been pre-trained on a large dataset was applied to learn the features of the leaf images and hence serve as the feature extractor to retrieve important information from the leaves. Lastly, the extracted features were trained and classified by using a machine learning method known as Logistic Regression.

The preprocessing was done by reducing the images to a uniform size of 256 * 256. Furthermore, the leaf images were paired with their corresponding patches to form a single image which would serve as a ready input for the cGAN model.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

A. cGAN-BASED METHOD OF BALANCING AND AUGMENTATION OF THE TRAINING DATA

Balancing data in deep learning applications like plant species classification is a critical area that has been studied and surveyed through the years and is driven by the necessity of a robust algorithm for effective classification and to aid it to generalize well. However, some essential elements to be considered are cost-effectiveness, user-friendliness, accuracy, and sensitivity. Data augmentation is the process of generating additional training data from the available existing data [25]. Typically, this is done by using annotation-preserving transformations on the input data, such as randomly rotating, deforming, or translating the image, scaling, flipping the images vertically and horizontally, and applying random zooming. Through the random nature of data augmentation, it can be used to potentially generate an 'infinite' amount of training data by augmenting the existing data. Model performance for tasks like classification, detection, and recognition of plant species can be improved using data augmentation which can overcome the problem of inadequate data and imbalanced distribution. In this research, we employed the use of conditional Generative Adversarial Networks (cGANs) to perform data augmentation.

Generative Adversarial Network(GAN) proposed by Goodfellow *et al.* [23] is a deep learning architecture for generating images, which composes of both a generator and a discriminator. The generator generates fake images from the input noise. The fake images created by the generator are given to the discriminator, which attempts to perform a binary classification to discern the genuineness of the images created by the generator. The discriminator is trained to maximize the probability where it can correctly discern the real images and the fake images. At the same time, the generator is trained to minimize log(1−D(G(z))) in Equation (1)[23].

$$L_{GANs}(G,D) = \mathbb{E}_y[\log D(\mathbf{y})] + \mathbb{E}_z[\log(1 - D(G(\mathbf{z}))] \quad \text{.....(1)}$$

Through this competitive and repetitive training process, the generator can generate fake images that are similar to the real original images. However, because it generates images based on input noise z, it is difficult to control the images created by the generator, and it experiences difficulties when generating high-resolution images. Because of this problem, Mirza and Osindero[26] introduced conditional Generative Adversarial Networks (cGANs), which extends GANs into a conditional model. This conditional form of GANs, namely cGANs[26] enables controllable image synthesis, allowing a user to synthesize images given various conditional inputs such as class labels, user sketches, or textual descriptions. In cGANs, the generator G and the discriminator D are conditioned on some extra information c. This is done by putting c as additional inputs to both G and D. cGANs provide additional controls on which kind of data are being generated, unlike the vanilla GANs which do not necessarily possess such controls. It makes cGANs popular for image synthesis and image editing applications. The structure of cGANs is illustrated as Fig. 2. The objective function which is the same as that of Pix2Pix [27] is represented by:

$$L_{Pix2Pix}(G,D) = \mathbb{E}_{x,y\sim p_{data}(x,y)}[logD(x,y)] + \mathbb{E}_{x\sim p_{data}(x),z\sim p_z(z)}[log(1 - D(x,G(x,z)))] \quad \text{.....(2)}$$

The cGAN model consists of a generator network that synthesizes leaf images and a discriminator network that distinguishes whether an image is the actual leaf image or leaf image produced by the generator network. Their architecture is described below:

1) The Generator

The generator network was based on U-Net [28] and consists of 15 layers having skip connections. The introduction of skip connections is strategically done between layer *i* and layer *n -i* when the total number of layers in the encoder-decoder structure was *n*, to avoid losing low-level information that is reduced by progressive down-sampling otherwise known as the vanishing gradient problem that could be encountered in training very deep convolutional networks. This method is effective in preserving the low-level information of the input in the output of the generator.

Further, the generator can synthesize more realistic leaf images from the learned features. When the generator is trained, the L1 distance (Equation (3)) between the ground truth image and the generated image is reflected in the objective function, as shown in Equation (4) [29]. Thereby, low-level information is strengthened so that leaf images can be generated that are clearer than conventional cGAN [27]. Moreover, as a means of preventing the blurring effect in the images created by the generator, the PatchGAN concept was applied to the discriminator. This is a method that attempts to classify the input images of the discriminator in N× area patches [27].

$$L_{L1}(G) = \mathbb{E}_{\mathbf{x},\mathbf{y},\mathbf{z}}\left[\left\|\mathbf{y} - G(\mathbf{x},\mathbf{z})\right\|_1\right] \quad \text{.....(3)}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

$$G^* = arg \min_G \max_D \mathcal{L}_{cGAN} + \lambda \mathcal{L}_{L1}(G) \text{ .....(4)}$$

The generator module receives conditioned leaf images of size 256×256×3 (height × width × channel) as input, and the feature maps are calculated with a 5 × 5 filter from the 1st to the 8th convolutional layers within the encoder. This special design was so that a separate pooling layer, such as max pooling, wouldn't have to be used. Special padding and stride of 2×2 were used to reduce the size of the feature map. The feature map, which is reduced to 1×1×512 by the 8th convolutional layer, is up-sampled through the 1st to 8th transposed convolutional layers within the decoder. To avoid losing low-level information, skip-connections were added between layer *i* and layer *n-i* when the total number of layers in the generator model is *n*. Figure 19 details the generator architecture of the cGAN model.

2) The Discriminator

The discriminator network is a Convolutional Neural Network consisting of four layers. It divides the input image into small patches to determine whether the input image is a real leaf image or a generated leaf image. It makes the generator represent the details of the leaf images better.

Figure 20 shows the architectural structure of the discriminator module of our cGAN model, which is used to discern whether the leaf images created by our generator module are genuine or fake. The input for the discriminator is a pair of images consisting of an image created by the generator and an image that uses extra information (fake image), or a pair of images consisting of a geometric center image and an image that uses extra information (real image), as shown in Equation (2). The input image is reduced to a 32×32×512 size as a result of going through the 4 convolutional layers. The reduced feature map is produced as the final value regarding whether the image is real or fake via the linear regression and sigmoid function of the fully connected layer.

3) The Combined cGAN Model

As training progressed, the two networks operated adversarially; the discriminator distinguishes the images more and more accurately, and it makes the generator synthesize more and more realistic leaf-like images. The loss function was also important to synthesize more realistic leaf-like images. We used cross-entropy for the adversarial loss of the generator network and the discriminator network. For the content loss of the generator network to synthesize images, we used L1 distance loss. The generator learns a mapping from the source image x and random noise vector z to the target image y, i.e., {x,z→y}. Discriminator discriminates the label of y|x, real or fake. Figure 3 shows the block diagram of the cGAN [23] model we utilized for data augmentation. The results are shown more clearly in section v of this paper.

B. FEATURE EXTRACTION USING A DEEP LEARNING MODEL AND CLASSIFICATION

Convolutional Neural Networks is now the state-of-the-art method for many tasks including classification, detection, localization, and segmentation [30]. Their architectures are most commonly applied to image analysis or other problems where shift-invariance or covariance is needed. Inspired by the fact that an object on an image can be shifted in the image and still be the same object, CNNs adopt convolutional kernels for the layer-wise affine transformation to capture this translational invariance. A 2D convolutional kernel $\boldsymbol{w}$ applied to a 2D image data $\boldsymbol{x}$ can be defined as

$$\boldsymbol{S}(i,j) = (\boldsymbol{X} * \boldsymbol{W})(i,j) = \sum_m \sum_n \boldsymbol{X}(m,n)\boldsymbol{W}(i-m, j-n), \text{ .....(5)}$$

where $\boldsymbol{S}(i,j)$ represents the output at position (i,j), $\boldsymbol{x}(m,n)$ is the value of the input $\boldsymbol{x}$ at position (m,n), $\boldsymbol{w}(i-m,j-n)$ is the parameter of kernel $\boldsymbol{w}$ at position $(i-m,j-n)$, and the summation is over all possible positions. An important variant of CNN is the residual network (ResNet), which incorporates skip-connections between layers. These modifications have shown great advantages in practice, aiding the optimization of these typically huge models.

1) Convolutional Layer:

This layer is responsible for the performance of convolution operations to learn features from the images. Here, a convolutional kernel slides along the image with a certain stride and outputs convolution plus a bias. The input to this layer could be either an RGB image or the output feature of another layer. Convolutional kernel means that given an input image when it is processed, the weighted average of pixels in a small area of the input image becomes each corresponding pixel in the output image, and the weight is defined by a function, and hence, they share weights to reduce parameters in the model. All the weights in this layer can be learned. This process can be expressed mathematically as

$$X_j^l = f\left(\sum_{i \in M_j} Y_i^{l-1} K_{ij}^l + b_j^l\right) \text{ .....(6)}$$

where Y_i^{l-1} is the output of the i *th* feature map in the $l-1$ layer, X_j^l is the input of the j *th* feature map in the l layer. K_{ij} and b_j^l are convolutional kernel and bias

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

in the *l* layer. f (.) is the activation function. Higher-level unique features can be identified through a series of increased convolution layers, hence, the need to go deeper.

2) Pooling Layer:

After convolution, we need to reduce the dimensionality of the image for further processing. This process is known as down sampling or a pooling operation. This process can be

$$X_j^l = f(\beta_j^l \cdot \text{down}_s(X_j^{l-1}) + d_j^l) \quad \text{.....(7)}$$

described as

where X_j^l represents the j *th* feature map in the l layer. β_j^l and d_j^l are the multiplicative factor and bias respectively. $\cdot\text{down}_s(.)$ represents an under-sampling function. Under-sampling can be done in many forms, some of which are average pooling, maximal pooling (max pool), minimal pooling operation, and so on. In our work, we employed max-pooling.

3) Fully Connected Layer:

Each neuron in the fully connected layer is connected to all neurons in the feature map of the previous layer, and the output can be expressed as:

$$h_{W,b}(x) = f(W^T + b) \quad \text{.....(8)}$$

where $h_{W,b}(x)$ is the output, and W represents the corresponding weights. The inputs to the fully connected layer are many features extracted from the previous layer. Each feature in the former layer represents different semantic information that is unique and important for further processing.

4) Loss Function:

The Loss function measures the disparity between the predicted output and the desired output. The network makes use of categorical cross-entropy for calculating loss, ϕ. The categorical cross-entropy function is given as:

$$\phi = -\frac{1}{K}\sum_{n=1}^{K}[y_n \log \hat{y}_n] \quad \text{.....(9)}$$

where y and $\hat{y}$ represent both the actual and the expected outputs respectively.

5) Activation Layer:

Non-linearity has proven to be an integral part of CNNs and makes them more powerful. CNNs must be able to take any input from $-\infty$ to $+\infty$, at the same time it should also be able to map it to an output that ranges between {0,1} or {-1,1} in some instances, thus, the need for activation function. Non-linearity is needed in activation functions because it aims at producing a nonlinear decision boundary via non-linear combinations of the weight and inputs in Deep Learning architectures.

Usually, a non-linear activation layer is applied immediately after each convolution layer in the training of CNNs. Various non-linear functions exist in the field of Deep Learning which are used to introduce non-linearity. Some of which are:

a) tanh:

This non-linearity takes the real-valued number to the range [−1,1], which is mathematically represented as:

$$tanh(x) = 2\sigma(2x) - 1 \quad \text{.....(10)}$$

b) Rectified Linear Unit(relu):

This increases the non-linear properties of the model without changing the receptive field of the convolution layer by changing all the negative values to 0. This can be represented as:

$$f(x) = \max(0, x) \quad \text{.....(11)}$$

c) Sigmoid:

This non-linearity takes the real-valued number to the range between [0, 1], ie. Mathematically it can be represented in the form

$$\sigma(x) = 1/(1+e^{-x}) \quad \text{.....(12)}$$

d) Softmax:

Classification can be done either by a binary classifier or a multi-class classifier. For binary classification, the activation function sigmoid is used whereas for multi-class classification, the softmax function is widely used. The softmax activation is utilized in the last dense layer to calculate the probability of predicted classes. The class with the highest probability is chosen as the output. The softmax function is shown mathematically by:

$$S(y_i) = \frac{e^{y_i}}{\sum_{j=1}^{K} e^{y_j}} \quad \text{.....(13)}$$

where e^{y_i} and e^{y_j} denote the probability belonging to the *i* and *j* categories respectively; *K* denotes the number of categories and it is initialized to sixteen in this work. The softmax function calculates the prediction of each category, $S(y_i)$.

The Deep Residual Network we used is 50 layers deep with a small receptive field of 7 × 7 in the input layer followed by a max-pooling layer of 3 × 3 kernel size. Figure 4 shows the additional identity mapping and Figure 5 the simplified block diagram of the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

Deep Residual Network we used for feature extraction while Table 3 details the parameters of the architecture of the network used. Figure 6 depicts a block diagram of the model used in the feature extraction and classification section.

C. LOGISTIC REGRESSION MODEL FOR CLASSIFICATION

Like all regressions analyses, Logistic Regression (LR) is a predictive analysis method. It is a qualitative statistical technique that is different from quantitative techniques. There are two types of logistic regression, binary and multinomial logistic regression. Whereby the binary logistic regression is applied to variables that are divided into two subgroups, resulting in only 0 or 1; for multinomial logistic regression, different outcomes are predicted. That is, it can display results in more than two groups and hence can be used to solve more complex problems than binary logistic regression. Multinomial LR system needs to find the relationship between input data and the output to create a model and generate weights for classification.

The Logistic regression model is of Equation (3), where $h\theta(x) = P(y=1|x, \theta)$ represents the probability of the output variable $y = 1$.

$$h_\theta(x) = g\left(\theta^T X\right) = \frac{1}{1+e^{-\theta^T X}} \quad \text{.....(14)}$$

Logistic regression is also known as logit regression or logistic model. It takes in independent features and returns the output as categorical output. The probability of occurrence of a categorical output can also be found by the LR model by fitting the features in the logistic curve. In our work, the features that were extracted by the Deep Residual Network model were the input to our LR model which went on to effectively classify them into the respective plant categories with very high accuracy. This is shown more vividly in Table 12 and a comparison with other models is displayed by Tables 13-15.

IV. DESCRIPTION OF EXPERIMENT

A. DATASET SETUP

The experiments in this study were carried out using about seven original datasets of plant leaves that have been made publicly available for research purposes. A brief description of each one used in the work is given in the subsequent subsections.

1) Flavia Dataset

The first is the flavia dataset consisting of 1703 plant leaf images of 32 different species most of them being common plants in the Yangtze Delta of the People's Republic of China. The images are of size 1600 × 1200. The images contain only blades, without petioles. The flavia dataset [12] being pre-processed is clean and has little noise. It is regarded as a fine-grained classification challenge where the task is to recognize 32 distinct species of plants using their leaf images. Table 2 gives a detailed description of the classes in this dataset.

This image dataset is quite small, having only an average of 65 images per class for a total of 1,360 images. A general rule of thumb when applying deep learning to computer vision tasks is to have 1,000 - 5,000 examples per class, so we are certainly at a huge deficit here. We call flavia classification a fine-grained classification task because all categories are very similar (i.e., species of plants). We can think of each of these categories as subcategories. The categories are certainly different, but share a significant amount of common structure (e.g. shape, color, vein, etc.) The result of our model on this dataset is represented in Table 6.

2) MalayaKew Dataset

The second dataset used is the MalayaKew (MK) Leaf dataset [18] which was collected at the Royal Botanic Gardens, Kew, England. It consists of scan-like images of leaves from 44 species classes. This dataset is very challenging as leaves from different species classes have a very similar appearance. Specifically, the D1 images of the MK dataset was used. It consists of segmented leaf images with size 256 * 256 pixels. Having 2288 training and 528 testing images respectively. The result of our model on this dataset, without adding synthetic data generated is represented by Table 10 while the result with the two combined datasets is represented by Table 11.

3) Swedish Dataset

This dataset, introduced by So¨derkvist [31], was captured as part of a joined leaf classification project between the Linkoping University and the Swedish Museum of Natural History. It contains images of isolated leaf scans on a plain background having 75 samples from each of 15 species of Swedish trees. This dataset is considered very challenging due to its high inter-species similarity. The result of our model on this dataset is represented in Table 8.

4) Middle European Woody Plants

This dataset, popularly known as MEW_2012 [32] is a collection of plant leaves from Europe. It contains native or frequently cultivated trees and shrubs of the Central Europe Region.

5) Folio Dataset

The folio dataset [33] is a standard leaf dataset used in plant recognition. The leaves were placed on a white background and then photographed. The pictures were taken in broad daylight to ensure optimum light intensity. It contains 32 different species taken from plants in the farm of the University of Mauritius and nearby locations. The result of our model on this dataset is represented in Table 9.

6) Amazon Forestry Dataset

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

The is the Peruvian Amazon Forestry Dataset [34] [35]. It is taken from the Allpahuayo-Mishana National Reserve in Peru contains 59,441 leaf images from ten timber tree species. The dataset is gathered in different excursions and conditions. Due to the unavailability of one of the classes on the internet at the time of conducting this research, only the available nine (9) classes were used for this research, therefore, not the entire dataset was used. The result of our model on this dataset is represented in Table 7.

7) LeafSnap Dataset

The final dataset used to make this research robust is the LeafSnap dataset. Published by Kumar et al. [14], it is a leaf dataset that includes images of leaves taken from two different sources and also segmentation produced automatically for them. They are divided into 23147 Laboratory photos, consisting of high-quality images of compact leaves, from the Smithsonian series and 7719 Field images, consisting of ordinary outdoor images provided with a mobile phone. These images contain a variety of blur, noise, lighting patterns, shadows, and so on. Each image is of size 800 × 600. Figures 1 and 2 show samples of some of the datasets.

B. EXPERIMENTAL SETUP

To implement and test the proposed method, a computer system has been employed for conducting the analysis processes. The computer system includes the following specifications: Windows 10, 64-bit, Intel Core i7-4720 CPU @ 2.60 GHz, RAM 32 GB and GPU Nvidia GeForce GTX 1050 4 GB dedicated memory, and Python 3.7 on Anaconda. The PC was used on a Linux-based DELL PowerEdge T640 Tower Server with CUDA-based video cards 4X 1080TI, each GPU Video memory is 11Gb, with a storage memory of 10TB Hard Drive and 3320GB SSD. Tables 4 and 5 detail the setup of the system used and the parameters used in running the model.

C. EVALUATION METRIC

We analyzed the proposed method for the accuracy, precision, recall and f1-score. The accuracy of the proposed plant recognition system has been computed via the following expression, which utilizes the numerical details encompassing True Positive (TP) (it is the number of leaf images that have been correctly identified), False Positive (FP) (it is a parameter for representation of the number of leaves that are incorrectly detected), True Negative (TN) (it is a parameter for representation of the number of leaf images that are correctly detected), and False Negative (FN) (it is a parameter for representation of the number of leaf images that are correctly recognized).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \text{.....(15)}$$

Also, precision and recall are defined as a measure for system evaluation. These concepts are calculated by Eqns. (16) and (17), respectively:

$$Precision = \frac{TP}{TP + FP} \text{.....(16)}$$

$$Recall = \frac{TP}{TP + FN} \text{.....(17)}$$

$$F1 = \frac{2 \times P \times R}{P + R} \text{.....(18)}$$

D. TRAINING DETAILS FOR THE cGAN

First, we prepared our dataset for training by generating the training data in the form of pairs of images and combining each pair of images into a single image file, hence making it ready for training by the cGAN model. Data were split into train, validation, and test sets randomly. This means creating a folder of images, each of which contains an X/Y pair of both original leaf images and their corresponding leaf patches. The X and Y image each occupy half of the full image in the set. Thus, they are the same size. By default, the images are assumed to be square (if they are not, they will be squashed into square input and output pairs).

In pix2pix[27], the testing mode is also set up to take image pairs just like in the training mode, where there is an X and a Y. We held out a test set of images from training, and afterward performed a comparison on the generated leaf images, Y to the known leaf images Y, as a way to visually evaluate the leaf images. We created an HTML page with a row for each sample containing the input, the output (generated Y), and the target (original Y) leaf images. We mostly followed the setting of [36] in training our cGAN model. More specifically, we use the Adam optimizer with $\beta1 = 0.5$ and $\beta2 = 0.999$.

For the Malayakew D1 dataset, 2816 training images constituted the training set, we trained for 3100 epochs with a learning rate of 0.0002, with random jitters and mirroring. We used a U-Net-based [28] generator, which has been described in subsection B of section III, with PatchGAN as the discriminator. We use a batch size of 1 and instance normalization [40] in place of Batch Normalization that is used in the original U-Net[28] architecture. All training images are loaded in their original sizes then cropped to 256 × 256 patches.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

E. TRAINING DETAILS FOR FEATURE EXTRACTION AND PLANT CLASSIFICATION

Table 5 details the parameters used in extracting the features and classification of the plant species. The training/test set split ratio was set to 75/25%. In as much as there is no fixed value for train/test split ratio, some previous experiments have shown some success around some particular value for training datasets depending on the size of the dataset. However, too few training samples can also cause the model to be underfitting irrespective of the size of the dataset.

Due to the large number of datasets to be tested and the prior success of the above ratio and ratios close to it on previous experiments [37][38][39], the above ratio was solely selected. However, in future experiments, multiple experiments with different train-test split values would be conducted with the sole purpose of determining the impact of train/test data split ratio on our method and the results bench-marked against state-of-the-art results on such classification.

The training data was used to extract the features that were further input to the Logistic Regression module for classification. The MalayaKew Dataset, whose augmented results are shown in Figures 8 & 9 was updated, now having a mixture of both the original and the augmented images in the newly created dataset was also processed by the feature extractor and classified. The results are tabulated in the subsequent tables, Tables 6-12 respectively.

The GridSearchCV class function of scikit-learn [40] was used to tune the parameters of the Logistic regression Classifier to find the best C, which stands for the strictness of the Logistic Regression classifier to determine what its optimal value is for the individual tasks. Each optimal value of C is stated on the tables that report the classification results(Tables 6-12).

V. RESULTS AND DISCUSSION

A. DATA AUGMENTATION RESULTS

We trained the cGAN for 3100 epochs using NVIDIA GeForce GTX 108 GPU. The training took an average of 4.0 seconds per epoch. During the training process, the discriminator and generator operate adversarial to each other. At first, the generator had difficulty making realistic simulator-like images, and the discriminator worked at distinguishing the real simulated images with images made by the generator with high accuracy. Therefore, the loss of discriminator was decreased. Then, to deceive the discriminator, the generator constructs more realistic simulator-like images. Thus, the loss of the discriminator increased, while that of the generator decreased gradually. As a result, the generator network can synthesize more and more realistic images of given leaf images as training progresses. Figures 8 and 9 show the results of the synthetic images of the leaves that were generated.

B. PLANT RECOGNITION RESULTS

The results of our work are carefully represented by figures (Figures 10-18), tables (Tables 6-12), and a chart below. Also, the comparison with other models is highlighted in Tables 13-15. Figures 10-17 clearly show the confusion matrices of the tested model on all of the datasets that were used in this work. Details reveal the actual classes versus the predicted classes, showing which classes the model was confused on and to what degree.

C. DISCUSSION

It is pertinent to note that the reason for the lower performance of the model on the MalayaKew dataset can be found in three very similar classes. Class 2 and 9 are highly similar, more of a fine-grained classification challenge; and also, class 42 was a problematic class to differentiate. This can be observed in Tables 10 & 11 that show both the results of the MK-D1 original and the MK-D1 original + synthetic datasets, though on the second dataset which was augmented the results were highly improved. The model had a 100% accuracy on the Swedish dataset, and the lowest accuracy being the LeafSnap leaf dataset with an 89.27% accuracy. It is clear why, because we used the LeafSnap dataset as it is, without adequate preprocessing due to time constraints.

Evaluating the quality of synthesized images is an open and difficult problem. In this study, we investigated the effectiveness of applying a conditional GAN's image-to-image translation model as a data augmentation tool to improve the performance of an automated plant classification system using plant leaf images. The accuracy of the conditional Generative Adversarial Network in producing augmented data and how very near the exact original images the model was able to produce the synthetic ones can be seen in Figures 8 & 9. From these figures, we noted that within the first 300 epochs of training, the network had difficulty generating images with a correct shape as close to the original images but began to make a better image with regards to the leaf shape afterward. Similarly, leaf veins became more distinct from the 800th epoch and got better from the 2400th epoch onwards.

From the results displayed, we observed that the percentage accuracy of the model on all the datasets combined produced an average of 96.1% and outperforms other research on the individual datasets. Also, it was observed that a combination of the original and synthetic MK-D1 datasets on testing the model produced a 3.0% increase in accuracy from the original dataset, proving that more data enables our Deep Learning models to perform better and generalize properly.

VI. CONCLUSION

Our research studied an extensive deep learning approach to classifying plants via leaf images. Mainly learning discriminative features from leaf images and applying a classifier for effective plant classification. We utilized conditional Generative Adversarial Networks to generate synthetic data where we got near accurate created leaf images to serve as augmentation for our MalayaKew-D1 dataset, Convolutional Neural Networks were then used to extract rich features from the leaves which were then fed into a Logistic Regression classifier for efficient classification of the plant species. From the experimental results, we justified that learning the features through CNN, specifically ResNet50 model can provide better feature representation for leaf images compared to hand-crafted features. Extensive experiments on each of the seven datasets used for this work demonstrate the superiority of our method compared with others, displaying an average performance of 96.1% for all the datasets, outperforming conventional solutions. Moreover, the result of the model on a combination of the original dataset and the synthetic MK-D1 dataset produced a 3.0% increase in the accuracy of the model's classification. The results in this paper suggest that conditional generative adversarial networks are a promising approach for data augmentation to tackle the problems of a lack of sufficient training data or uneven class balance that could be found within datasets in Deep Learning tasks. More work could be done to augment other plant datasets to enable robust Deep Learning architectures to have more data to learn from, to generalize well.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (No.U1813222, No.42075129), Tianjin Natural Science Foundation (No.18JCYBJC16500) and Key Research and Development Project from Hebei Province(No.19210404D, No.20351802D).

REFERENCES

[1] K. J. Willis (Ed.), "State of the World's Plants 2017. Report," 2017.

[2] A. Kadir, L. E. Nugroho, A. Susanto, and P. I. Santosa, "Leaf Classification Using Shape, Color, and Texture Features," *Int. J. Comput. Trends Technol.*, Nov. 2013, Accessed: Dec. 18, 2020. [Online]. Available: http://arxiv.org/abs/1401.4447.

[3] J. S. Cope, D. Corney, J. Y. Clark, P. Remagnino, and P. Wilkin, "Plant species identification using digital morphometrics: A review," *Expert Systems with Applications*. 2012, doi: 10.1016/j.eswa.2012.01.073.

[4] N. Ahmed, U. Ghani Khan, and S. Asif, "AN AUTOMATIC LEAF BASED PLANT IDENTIFICATION SYSTEM," *Sci.Int.(Lahore)*, 2016.

[5] T. Jin, X. Hou, P. Li, and F. Zhou, "A novel method of automatic plant species identification using sparse representation of leaf tooth features," *PLoS One*, 2015, doi: 10.1371/journal.pone.0139482.

[6] J. C. Neto, G. E. Meyer, D. D. Jones, and A. K. Samal, "Plant species identification using Elliptic Fourier leaf shape analysis," *Comput. Electron. Agric.*, 2006, doi: 10.1016/j.compag.2005.09.004.

[7] J. Chaki and R. Parekh, "Plant Leaf Recognition using Shape based Features and Neural Network classifiers," *Int. J. Adv. Comput. Sci. Appl.*, 2011, doi: 10.14569/ijacsa.2011.021007.

[8] J. X. Du, X. F. Wang, and G. J. Zhang, "Leaf shape based plant species recognition," *Appl. Math. Comput.*, 2007, doi: 10.1016/j.amc.2006.07.072.

[9] S. Mouine, I. Yahiaoui, and A. Verroust-Blondet, "Advanced shape context for plant species identification using leaf image retrieval," 2012, doi: 10.1145/2324796.2324853.

[10] X. Y. Xiao, R. Hu, S. W. Zhang, and X. F. Wang, "HOG-based approach for leaf classification," 2010, doi: 10.1007/978-3-642-14932-0_19.

[11] A. Aakif and M. F. Khan, "Automatic classification of plants based on their leaves," *Biosyst. Eng.*, vol. 139, pp. 66–75, Nov. 2015, doi: 10.1016/j.biosystemseng.2015.08.003.

[12] S. G. Wu, F. S. Bao, E. Y. Xu, Y.-X. Wang, Y.-F. Chang, and Q.-L. Xiang, "A Leaf Recognition Algorithm for Plant Classification Using Probabilistic Neural Network," in *2007 IEEE International Symposium on Signal Processing and Information Technology*, Dec. 2007, pp. 11–16, doi: 10.1109/ISSPIT.2007.4458016.

[13] D. Hall, C. McCool, F. Dayoub, N. Sünderhauf, and B. Upcroft, "Evaluation of features for leaf classification in challenging conditions," 2015, doi: 10.1109/WACV.2015.111.

[14] N. Kumar *et al.*, "LeafSnap: A computer vision system for automatic plant species identification," 2012, doi: 10.1007/978-3-642-33709-3_36.

[15] M. G. Larese, R. Namías, R. M. Craviotto, M. R. Arango, C. Gallo, and P. M. Granitto, "Automatic classification of legumes using leaf vein image features," *Pattern Recognit.*, vol. 47, no. 1, pp. 158–168, 2014, doi: 10.1016/j.patcog.2013.06.012.

[16] P. Wilf, S. Zhang, S. Chikkerur, S. A. Little, S. L. Wing, and T. Serre, "Computer vision cracks the leaf code," *Proc. Natl. Acad. Sci. U. S. A.*, 2016, doi: 10.1073/pnas.1524473113.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," 2012, doi:

10.1061/(ASCE)GT.1943-5606.0001284.

[18] S. H. Lee, C. S. Chan, P. Wilkin, and P. Remagnino, "Deep-plant: Plant identification with convolutional neural networks," *Proc. - Int. Conf. Image Process. ICIP*, vol. 2015-Decem, pp. 452–456, 2015, doi: 10.1109/ICIP.2015.7350839.

[19] C. Zhang, P. Zhou, C. Li, and L. Liu, "A convolutional neural network for leaves recognition using data augmentation," 2015, doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.318.

[20] H. Goëau, P. Bonnet, and A. Joly, "Plant identification in an open-world (LifeCLEF 2016)," 2016.

[21] G. L. Grinblat, L. C. Uzal, M. G. Larese, and P. M. Granitto, "Deep learning for plant identification using vein morphological patterns," *Comput. Electron. Agric.*, 2016, doi: 10.1016/j.compag.2016.07.003.

[22] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Oct. 2015, vol. 07-12-June, pp. 1–9, doi: 10.1109/CVPR.2015.7298594.

[23] I. Goodfellow *et al.*, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020, doi: 10.1145/3422622.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2016, doi: 10.1109/CVPR.2016.90.

[25] L. Sharon, Y., "Automating Data Augmentation: Practice, Theory and New Direction," *SAIL Blog*, 2020. https://ai.stanford.edu/blog/data-augmentation/ (accessed May 25, 2021).

[26] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," pp. 2–3, Nov. 2014, Accessed: Oct. 25, 2021. [Online]. Available: https://arxiv.org/abs/1411.1784v1.

[27] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5967–5976, Nov. 2016, Accessed: May 29, 2021. [Online]. Available: http://arxiv.org/abs/1611.07004.

[28] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9351, pp. 234–241, May 2015, Accessed: Sep. 09, 2021. [Online]. Available: https://arxiv.org/abs/1505.04597v1.

[29] M. B. Lee, Y. H. Kim, and K. R. Park, "Conditional generative adversarial network-based data augmentation for enhancement of iris recognition accuracy," *IEEE Access*, vol. 7, pp. 122134–122152, 2019, doi: 10.1109/ACCESS.2019.2937809.

[30] J. Schlemper *et al.*, "Attention gated networks: Learning to leverage salient regions in medical images," *Med. Image Anal.*, vol. 53, pp. 197–207, Apr. 2019, doi: 10.1016/J.MEDIA.2019.01.012.

[31] O. J. O. Söderkvist, "Computer Vision Classification of Leaves from Swedish Trees," Linkoping University, Linkoping, Sweden, 2001.

[32] P. Novotný and T. Suk, "Leaf recognition of woody species in Central Europe," *Biosyst. Eng.*, vol. 115, no. 4, pp. 444–452, Aug. 2013, doi: 10.1016/j.biosystemseng.2013.04.007.

[33] T. Munisami, M. Ramsurn, S. Kishnah, and S. Pudaruth, "Plant Leaf Recognition Using Shape Features and Colour Histogram with K-nearest Neighbour Classifiers," *Procedia Comput. Sci.*, vol. 58, pp. 740–747, 2015, doi: 10.1016/j.procs.2015.08.095.

[34] G. Vizcarra, D. Bermejo, A. Mauricio, R. Zarate Gomez, and E. Dianderas, "The Peruvian Amazon forestry dataset: A leaf image classification corpus," *Ecol. Inform.*, vol. 62, p. 101268, May 2021, doi: 10.1016/J.ECOINF.2021.101268.

[35] "Amazon Leaf Dataset." http://teledeteccion.iiap.gob.pe/dataset/en/ (accessed May 25, 2021).

[36] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," *arXiv*, vol. 2017-Janua, pp. 5967–5976, Nov. 2016, Accessed: May 29, 2021. [Online]. Available: http://arxiv.org/abs/1611.07004.

[37] G. Van Horn *et al.*, "The iNaturalist Species Classification and Detection Dataset," Accessed: Sep. 07, 2021. [Online]. Available: www.inaturalist.org.

[38] Q. H. Nguyen *et al.*, "Influence of data splitting on performance of machine learning models in prediction of shear strength of soil," *Math. Probl. Eng.*, vol. 2021, 2021, doi: 10.1155/2021/4832864.

[39] A. Rosebrock, *Deep Learning for Computer Vision with Python: Practitioner Bundle*, 3rd ed. Pyimagesearch, 2019.

[40] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, vol. 12, no. 85, pp. 2825–2830, 2011, Accessed: Oct. 30, 2021. [Online]. Available: http://jmlr.org/papers/v12/pedregosa11a.html.

[41] C. A. Priya, T. Balasaravanan, and A. S. Thanamani, "An efficient leaf recognition algorithm for plant classification using support vector machine," 2012, doi: 10.1109/ICPRIME.2012.6208384.

[42] C.-Y. Gwo and C.-H. Wei, "Plant Identification Through Images: Using Feature Extraction of

Key Points on Leaf Contours," *Appl. Plant Sci.*, vol. 1, no. 11, p. 1200005, Nov. 2013, doi: 10.3732/apps.1200005.

[43] S. Lavania and P. S. Matey, "Leaf recognition using contour based edge detection and SIFT algorithm," in *2014 IEEE International Conference on Computational Intelligence and Computing Research*, Dec. 2014, pp. 1–4, doi: 10.1109/ICCIC.2014.7238345.

[44] S. Anubha Pearline, V. Sathiesh Kumar, and S. Harini, "A study on plant recognition using conventional image processing and deep learning approaches," *J. Intell. Fuzzy Syst.*, vol. 36, no. 3, pp. 1997–2004, 2019, doi: 10.3233/JIFS-169911.

[45] M. Keivani, J. Mazloum, E. Sedaghatfar, and M. B. Tavakoli, "Automated analysis of leaf shape, texture, and color features for plant classification," *Trait. du Signal*, vol. 37, no. 1, pp. 17–28, 2020, doi: 10.18280/ts.370103.

[46] P. Pawara, E. Okafor, L. Schomaker, and M. Wiering, "Data augmentation for plant classification," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10617 LNCS, pp. 615–626, 2017, doi: 10.1007/978-3-319-70353-4_52.

[47] D. G. Tsolakidis, D. I. Kosmopoulos, and G. Papadourakis, "Plant leaf recognition using Zernike moments and histogram of oriented gradients," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8445 LNCS, pp. 406–417, doi: 10.1007/978-3-319-07064-3_33.

[48] X. Zhang, W. Zhao, H. Luo, L. Chen, J. Peng, and J. Fan, "Plant recognition via leaf shape and margin features," *Multimed. Tools Appl.*, vol. 78, no. 19, pp. 27463–27489, Oct. 2019, doi: 10.1007/s11042-019-07846-0.

[49] D. Bisen, "Deep convolutional neural network based plant species recognition through features of leaf," *Multimed. Tools Appl.*, vol. 80, no. 4, pp. 6443–6456, Feb. 2021, doi: 10.1007/s11042-020-10038-w.

[50] A. Sujith and R. Neethu, "Classification of plant leaf using shape and texture features," in *Lecture Notes in Networks and Systems*, 2021, vol. 145, pp. 269–282, doi: 10.1007/978-981-15-7345-3_22.

[51] N. Goyal, N. Kumar, and Kapil, "On solving leaf classification using linear regression," *Multimed. Tools Appl.*, vol. 80, no. 3, pp. 4533–4551, Sep. 2020, doi: 10.1007/s11042-020-09899-y.

[52] Y. Zhang, J. Cui, Z. Wang, J. Kang, and Y. Min, "Leaf image recognition based on bag of features," *Appl. Sci.*, vol. 10, no. 15, p. 5177, Aug. 2020, doi: 10.3390/app10155177.

[53] M. Turkoglu and D. Hanbay, "Leaf-based plant species recognition based on improved local binary pattern and extreme learning machine," *Physica A*, vol. 527, p. 121297, 2019, doi: 10.1016/j.physa.2019.121297.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

TABLE I
SPECIFICATION OF THE GENERAL LEAF FEATURES USED IN CLASSIFICATION

Main features	Sub-features
Shape	• Diameter • Aspect ratio • Rectangularity • Perimeter • Degree of the leaf tip • Roundness/ circularity • Slimness
Color	Mean and std-dev of Red, Green and Blue (RGB) channels
Texture	• Contrast • Correlation • Entropy • Homogeneity

TABLE 2
DETAILED DESCRIPTION OF THE FLAVIA LEAF DATASET

Scientific Name	Common Name(s)	Testing Sample
Phyllostachys edulis (Carr.) Houz	Pubescent Bamboo	50
Aesculus chinensis	Chinese Horse Chestnut	50
Berberis anhweiensis Ahrendt	Anhui Barberry	50
Cercis chinensis	Chinese Redbud	50
Indigofera tinctoria L.	True Indigo	50
Phoebe nanmu (Oliv.) Gamble	Nanmu	50
Kalopanax septemlobus (Thunb. ex A.Murr.)	Koidz. Castor Aralia	50
Cinnamomum japonicum Sieb.	Chinese Cinnamon	50
Koelreuteria paniculata Laxm.	Golden Rain Tree	50
Ilex macrocarpa Oliv.	Big-fruited Holly	50
Pittosporum tobira (Thunb.) Ait.	f. Japanese Cheesewood	50
Chimonanthus praecox L.	Wintersweet	50
Cinnamomum camphora (L.)	J. Presl Camphortree	50
Viburnum awabuki K.Koch	Japanese Arrowwood	50
Osmanthus fragrans Lour.	Sweet Osmanthus	50
Cedrus deodara (Roxb.) G.	Don Deodar	50
Ginkgo biloba L.	Ginkgo, Maidenhair Tree	50
Lagerstroemia indica (L.) Pers.	Crape myrtle, Crepe myrtle	50
Nerium oleander L.	Oleander	50
Podocarpus macrophyllus (Thunb.)	Sweet Yew Plum Pine	50
Prunus serrulata Lindl. var. lannesiana auct.	Japanese Flowering Cherry	50
Ligustrum lucidum Ait. f.	Glossy Privet	50
Tonna sinensis M. Roem.	Chinese Toon	50
Prunus persica (L.)	Batsch Peach	50
Manglietia fordiana Oliv.	Ford Woodlotus	50
Acer buergerianum Miq.	Trident Maple	50
Mahonia bealei (Fortune) Carr.	Beale's Barberry	50
Magnolia grandiflora L.	Southern Magnolia	50
Populus × Canadensis Moench	Canadian Poplar	50
Liriodendron Chinese (Hemsl.) Sarg.	Chinese Tulip Tree	50
Citrus reticulata Blanco	Tangerine	50

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

TABLE 3

CNN ARCHITECTURE USED FOR SELECTION OF LEAF FEATURE SUBSETS.

FIRST, SECOND AND THIRD ROW INDICATES LAYER NAME, NUMBER OF CHANNELS AND FILTER SIZE RESPECTIVELY.

conv1	**pool1**	**conv2**	**pool2**	**conv3**	**conv4**	**conv5**	**pool5**	**fc6**	**fc7**	**fc8**
96	96	256	256	384	384	256	256	4096	4096	1000
11 × 11	3 ×3	5 × 5	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3

TABLE 4

CONFIGURATION OF THE MACHINE USED.

Name	Parameter
Memory	32GB
Processor	Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz
Server model	DELL PowerEdge T640 Tower Server
Graphics	CUDA based video cards 4X 1080TI; GPU Video memory of 11Gb
OS	Linux
Language	Python 3
Framework	Pytorch, Tensorflow and Keras

TABLE 5

HYPERPARAMETERS USED FOR FEATURE EXTRACTION AND CLASSIFICATION.

Name	Parameter
Solver type	Adam
LR GridSearchCV	3
Batch Size	16
Input size	224x224
Feature extraction buffer size	1000
Train/Test split	0.75/0.25

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

TABLE 6

TABULATED PRECISION, RECALL, F1-SCORE VALUES FOR TEST SET ON FLAVIA DATASET WITH BEST HYPERPARAMETER C = 1

Class	Precision	Recall	f1-score	Support
Anhui_Barberry	1.00	1.00	1.00	16
Beales_barberry	1.00	1.00	1.00	13
Big-fruited_Holly	1.00	1.00	1.00	13
Canadian_poplar	1.00	1.00	1.00	17
Chinese_Toon	0.93	1.00	0.96	13
Chinese_cinnamon	1.00	1.00	1.00	14
Chinese_horse_chestnut	1.00	1.00	1.00	14
Chinese_redbud	1.00	1.00	1.00	21
Chinese_tulip_tree	1.00	1.00	1.00	13
Crape_myrtle	1.00	1.00	1.00	14
Ford_Woodlotus	1.00	1.00	1.00	4
Glossy_Privet	1.00	1.00	1.00	17
Japan_Arrowwood	1.00	1.00	1.00	15
Japanese_Flowering_ Cherry	1.00	1.00	1.00	18
Japanese_cheesewood	1.00	1.00	1.00	15
Japanese_maple	1.00	1.00	1.00	13
Nanmu	1.00	1.00	1.00	20
camphortree	0.94	1.00	0.97	21
castor_aralia	1.00	1.00	1.00	12
deodar	1.00	1.00	1.00	24
ginkgo,maidenhair_tree	1.00	1.00	1.00	13
goldenrain_tree	1.00	1.00	1.00	14
oleander	1.00	1.00	1.00	14
peach	0.94	0.94	0.94	18
pubescent_bamboo	1.00	1.00	1.00	17
southern_magnolia	1.00	1.00	1.00	12
sweet_osmanthus	1.00	0.92	0.96	12
tangerine	1.00	1.00	1.00	7
trident_maple	1.00	1.00	1.00	14
true_indigo	1.00	1.00	1.00	23
wintersweet	1.00	1.00	1.00	9
Yew_plum_pine	1.00	1.00	1.00	17
macro avg	**1.00**	**1.00**	**1.00**	**477**
weighted avg	**1.00**	**1.00**	**1.00**	**477**
total_score	**0.995807**			

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

TABLE 7

TABULATED PRECISION, RECALL, F1-SCORE VALUES FOR TEST SET ON AMAZON FORESTRY DATASET WITH BEST HYPERPARAMETER C = 0.001

Class	Precision	Recall	f1-score	Support
OtobaGlyc	0.99	0.98	0.99	1362
anibaRosa	0.98	0.98	0.98	1560
cedrelaOdora	0.99	0.98	0.99	1412
cedrelingaCate	0.98	0.99	0.99	1361
dipteryxMicre	0.99	1.00	1.00	1409
simarubaAmara	0.99	0.99	0.99	1352
swieteniaMacrophylla	0.99	0.99	0.99	1633
virolaFlexuosa	0.99	0.99	0.99	1068
virolaPavonis	0.99	1.00	0.99	1886
accuracy			**0.99**	**13043**
macro avg	**0.99**	**0.99**	**0.99**	**13043**
weighted avg	**0.99**	**0.99**	**0.99**	**13043**
total_score	**0.988652**			

TABLE 8

TABULATED PRECISION, RECALL, F1-SCORE VALUES FOR TEST SET ON SWEDISH DATASET WITH BEST HYPERPARAMETER C = 0.001

Class	Precision	Recall	f1-score	Support
Acer	1.00	1.00	1.00	19
Alnus incana	1.00	1.00	1.00	20
Betula pubescens	1.00	1.00	1.00	12
Fagus silvatica	1.00	1.00	1.00	15
Populus	1.00	1.00	1.00	18
Populus tremula	1.00	1.00	1.00	22
Quercus	1.00	1.00	1.00	16
Salix alba ‘Sericea’	1.00	1.00	1.00	15
Salix aurita	1.00	1.00	1.00	27
Salix sinerea	1.00	1.00	1.00	14
Sorbus aucuparia	1.00	1.00	1.00	19
Sorbus intermedia	1.00	1.00	1.00	24
Tilia	1.00	1.00	1.00	20
Ulmus carpinifolia	1.00	1.00	1.00	21
Ulmus glabra	1.00	1.00	1.00	20
accuracy			**1.00**	**282**
macro avg	**1.00**	**1.00**	**1.00**	**282**
weighted avg	**1.00**	**1.00**	**1.00**	**282**
total_score	**1.0**			

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

TABLE 9

TABULATED PRECISION, RECALL, F1-SCORE VALUES FOR TEST SET ON FOLIO DATASET WITH BEST HYPERPARAMETER C = 0.001

Class	Precision	Recall	f1-score	Support
ashanti blood	1.00	1.00	1.00	2
barbados cherry	1.00	1.00	1.00	9
beaumier du perou	1.00	1.00	1.00	9
betel	1.00	1.00	1.00	8
bitter orange	1.00	1.00	1.00	8
caricature plant	1.00	1.00	1.00	4
chinese guava	1.00	1.00	1.00	3
chocolate tree	1.00	1.00	1.00	5
chrysanthemum	1.00	1.00	1.00	4
coeur demoiselle	1.00	1.00	1.00	8
coffee	1.00	1.00	1.00	4
croton	1.00	1.00	1.00	2
duranta gold	1.00	1.00	1.00	3
eggplant	1.00	1.00	1.00	7
ficus	1.00	1.00	1.00	4
fruitcitere	1.00	1.00	1.00	3
geranium	1.00	1.00	1.00	5
guava	0.94	1.00	1.00	4
hibiscus	1.00	1.00	1.00	9
jackfruit	1.00	1.00	1.00	5
ketembilla	1.00	0.89	0.94	9
lychee	1.00	1.00	1.00	6
mulberry leaf	1.00	1.00	1.00	4
papaya	1.00	1.00	1.00	6
pimento	0.75	1.00	0.86	3
pomme jacquot	1.00	1.00	1.00	2
rose	1.00	1.00	1.00	7
star apple	1.00	0.80	0.89	5
sweet olive	0.75	1.00	0.86	3
sweet potato	1.00	1.00	1.00	2
thevetia	1.00	1.00	1.00	5
vieux garcon	1.00	1.00	1.00	2
macro avg	**0.98**	**0.99**	**0.99**	**160**
weighted avg	**0.99**	**0.99**	**0.99**	**160**
total_score	**0.9875**			

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

TABLE 10

TABULATED PRECISION, RECALL, F1-SCORE VALUES FOR TEST SET ON ORIGINAL MALAYAKEW-D1 DATASET WITH BEST HYPERPARAMETER C = 1

Class	Precision	Recall	f1-score	Support
Classl	0.82	1.00	0.90	14
Class2	0.40	0.24	0.30	17
Class3	0.94	1.00	0.97	17
Class4	1.00	0.88	0.93	16
Class5	1.00	1.00	1.00	13
Class6	1.00	0.92	0.96	13
Class7	0.95	1.00	0.98	20
Class8	1.00	1.00	1.00	12
Class9	0.46	0.35	0.40	17
Classl0	1.00	1.00	1.00	12
Classll	1.00	1.00	1.00	18
Class12	0.88	0.88	0.88	16
Class13	1.00	1.00	1.00	14
Class14	1.00	0.95	0.97	19
Class15	0.94	1.00	0.97	16
Class16	1.00	1.00	1.00	14
Class17	1.00	0.93	0.97	15
Class18	1.00	1.00	1.00	16
Class19	1.00	1.00	1.00	15
Class20	1.00	1.00	1.00	14
Class21	1.00	1.00	1.00	16
Class22	0.74	1.00	0.85	14
Class23	0.82	0.82	0.82	17
Class24	0.86	0.83	0.84	23
Class25	0.94	1.00	0.97	15
Class26	0.92	0.92	0.92	25
Class27	0.77	0.91	0.83	11
Class28	0.86	1.00	0.92	12
Class29	0.69	0.82	0.75	11
Class30	0.88	0.94	0.91	16
Class31	0.82	0.93	0.87	15
Class32	1.00	0.42	0.59	19
Class33	1.00	1.00	1.00	19
Class34	1.00	1.00	1.00	15
Class35	0.94	1.00	0.97	16
Class36	0.90	0.86	0.88	21
Class37	0.67	0.91	0.77	11
Class38	0.71	0.80	0.75	15
Class39	0.80	0.89	0.84	18
Class40	0.93	1.00	0.96	13
Class41	0.88	1.00	0.94	22
Class42	0.86	0.57	0.69	21
Class43	0.95	1.00	0.97	18
Class44	0.86	0.92	0.89	13
accuracy			**0.89**	**704**
macro avg	**0.89**	**0.90**	**0.89**	**704**
weighted avg	**0.89**	**0.89**	**0.89**	**704**
total_score	**0.89347**			

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

TABLE 11

TABULATED PRECISION, RECALL, F1-SCORE VALUES FOR TEST SET ON MALAYAKEW-D1 + SYNTHETIC DATASET WITH BEST HYPERPARAMETER C = 0.1

Class	Precision	Recall	f1-score	Support
Classl	0.93	0.91	0.92	75
Class2	0.76	0.81	0.78	58
Class3	0.99	0.91	0.95	77
Class4	0.95	0.98	0.96	54
Class5	0.91	0.96	0.93	74
Class6	0.94	0.97	0.96	70
Class7	0.97	0.93	0.95	74
Class8	0.89	0.99	0.94	68
Class9	0.80	0.69	0.74	65
Classl0	1.00	0.97	0.99	70
Classll	0.98	0.95	0.97	65
Class12	0.88	0.94	0.91	65
Class13	0.96	0.97	0.97	71
Class14	0.94	0.99	0.96	76
Class15	0.94	0.99	0.96	75
Class16	0.95	0.98	0.97	63
Class17	0.97	0.98	0.98	64
Class18	0.94	0.99	0.96	73
Class19	1.00	0.98	0.99	63
Class20	0.93	1.00	0.96	65
Class21	0.94	0.96	0.95	69
Class22	0.94	0.93	0.93	69
Class23	0.95	0.84	0.89	70
Class24	0.88	0.90	0.89	62
Class25	0.89	0.97	0.93	60
Class26	0.90	0.95	0.92	58
Class27	0.92	0.95	0.94	64
Class28	0.96	0.96	0.96	70
Class29	0.97	0.98	0.98	65
Class30	0.93	0.93	0.93	67
Class31	0.92	0.82	0.87	74
Class32	0.87	0.77	0.82	53
Class33	0.97	0.98	0.98	65
Class34	1.00	0.93	0.96	56
Class35	0.91	0.98	0.94	61
Class36	0.99	0.96	0.97	78
Class37	0.99	0.88	0.93	92
Class38	0.89	0.86	0.87	69
Class39	0.94	0.93	0.93	67
Class40	0.93	0.97	0.95	66
Class41	0.98	1.00	0.99	55
Class42	0.86	0.91	0.88	65
Class43	0.98	0.97	0.98	64
Class44	0.88	0.84	0.86	69
accuracy			**0.93**	**2953**
macro avg	**0.93**	**0.93**	**0.93**	**2953**
weighted avg	**0.93**	**0.93**	**0.93**	**2953**
total_score	**0.9333**			

TABLE 12

SUCCESS RATES OF OUR MODEL ON VARIOUS DATA SETS.

THE COLUMN LABELED #S CONTAINS THE NUMBERS OF SPECIES(CLASSES).

Dataset	***#s***	**Precision**	**Recall**	**F1-Score**	**Accuracy**
MEW2012	119	0.99	0.98	0.98	0.9872
Flavia	32	0.99	0.99	0.99	0.9958
MalayaKew(D1)	44	0.92	0.91	0.91	0.8935
MK(D1) + Synthetic dataset	44	0.93	0.93	0.93	0.9333
Folio	32	0.98	0.99	0.99	0.9875
Amazon forest	9	0.99	0.99	0.99	0.9887
LeafSnap	185	0.92	0.90	0.90	0.8927
Swedish	15	1.00	1.00	1.00	1.0000

TABLE 13

COMPARISON OF THE PROPOSED TECHNIQUE WITH THE EXISTING PREVALENT APPROACHES ON THE FLAVIA DATASET.

Technique	**Classifier**	**Accuracy (%)**
Wu et al., 2007 [12]	PNN	90.30
Priya et al., 2012 [41]	KNN, SVM	94.50
Gwo and Wei, 2013 [42]	Bayesian	92.70
Kadir et al., 2013 [2]	PNN	93.75
Lavania and Matey, 2014 [43]	SIFT	87.50
Aakif and Khan, 2015 [11]	ANN	96.00
Anubha Pearline et al., 2019 [44]	CNN + LR	96.38
Keivani et al ,. 2020 [45]	Decision Tree	98.58
Proposed Technique	**DL + LR**	**99.58**

TABLE 14

COMPARISON OF THE PROPOSED TECHNIQUE WITH THE EXISTING PREVALENT APPROACHES ON THE FOLIO DATASET.

Technique	**Classifier**	**Accuracy (%)**
Munisami et al., 2015 [33]	KNN	87.20
Anubha Pearline et al., 2019 [44]	CNN + LR	96.53
Keivani et al., 2020 [45]	Decision Tree	90.02
Pawara et al., 2017 [46]	CNN	97.70
Proposed Technique	**DL + LR**	**98.75**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

TABLE 15

COMPARISON OF THE PROPOSED TECHNIQUE WITH THE EXISTING PREVALENT APPROACHES ON THE SWEDISH DATASET.

Technique	**Classifier**	**Accuracy (%)**
Tsolakidis et al., 2014 [47]	SVM	98.13
Zhang et al., 2019 [48]	SVM	96.14
Bisen, 2021 [49]	CNN	97.00
Sujith and Neethu, 2021 [50]	PHOG, LBP GLCM	98.23
Goyal et al., 2020 [51]	LDA	94.24
Anubha Pearline et al., 2019 [44]	CNN + LR	99.41
Zhang et al., 2020 [52]	SVM	97.93
Turkoglu and Hanbay, 2019 [53]	LBP and ELM	99.46
Proposed Technique	**DL + LR**	**100.00**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

FIGURE 1. **Sample leaf images of the flavia dataset.**

FIGURE 2. **Sample leaf images of the MalayaKew dataset** [18].

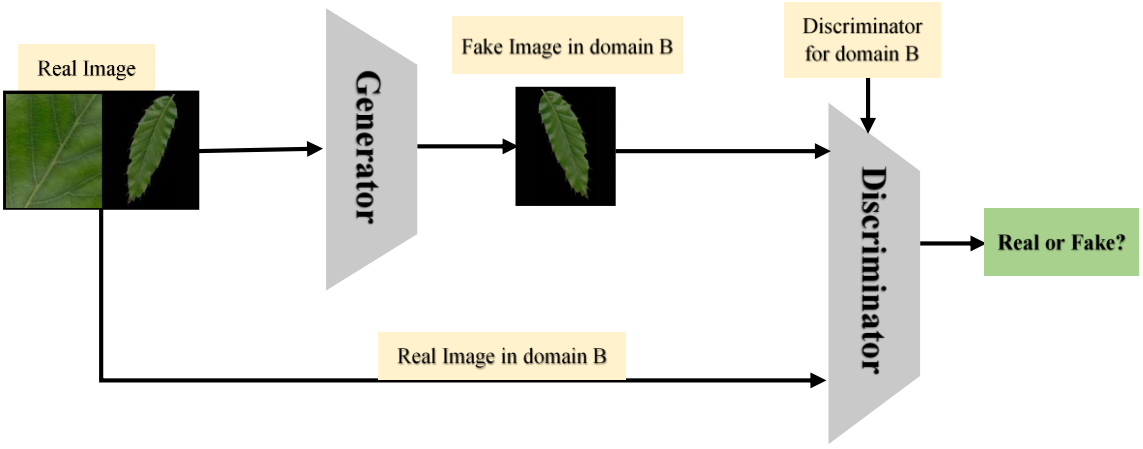

FIGURE 3. **Block diagram of the cGAN model.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

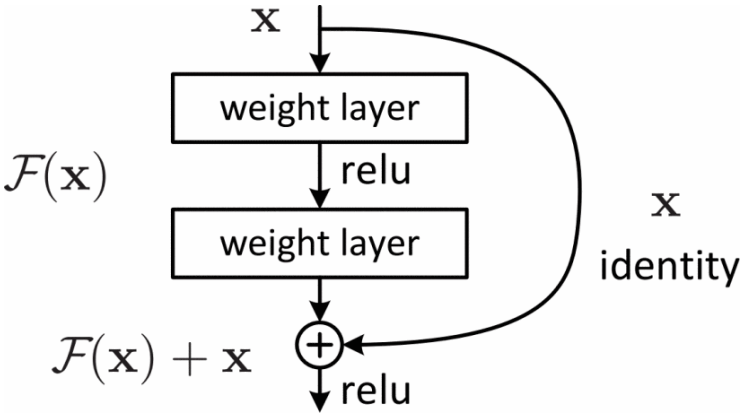

FIGURE 4. The residual identity mapping.

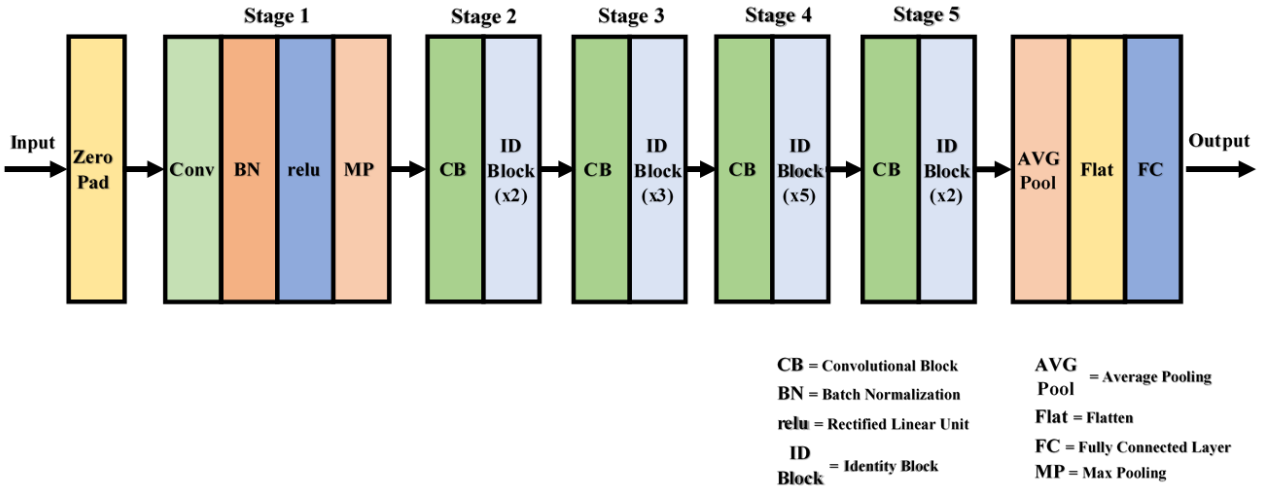

FIGURE 5. A simplified block diagram of the Residual Network architecture with 50 layers.

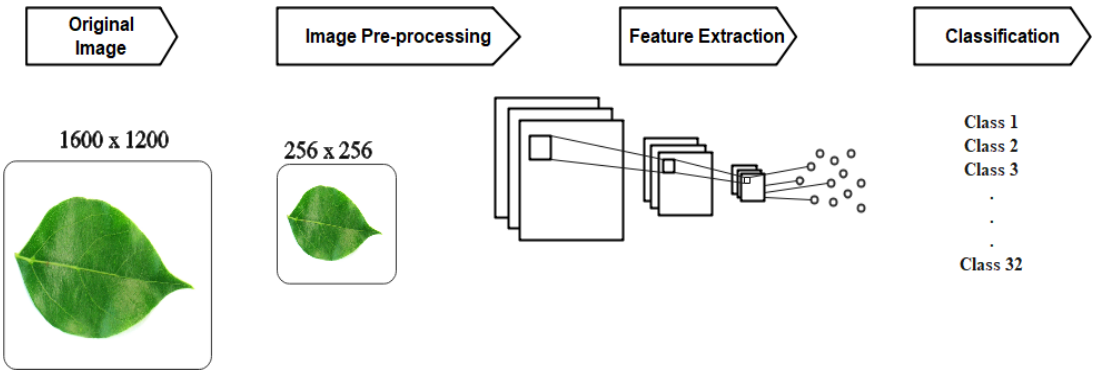

FIGURE 6. Layout of classification of the leaf images.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

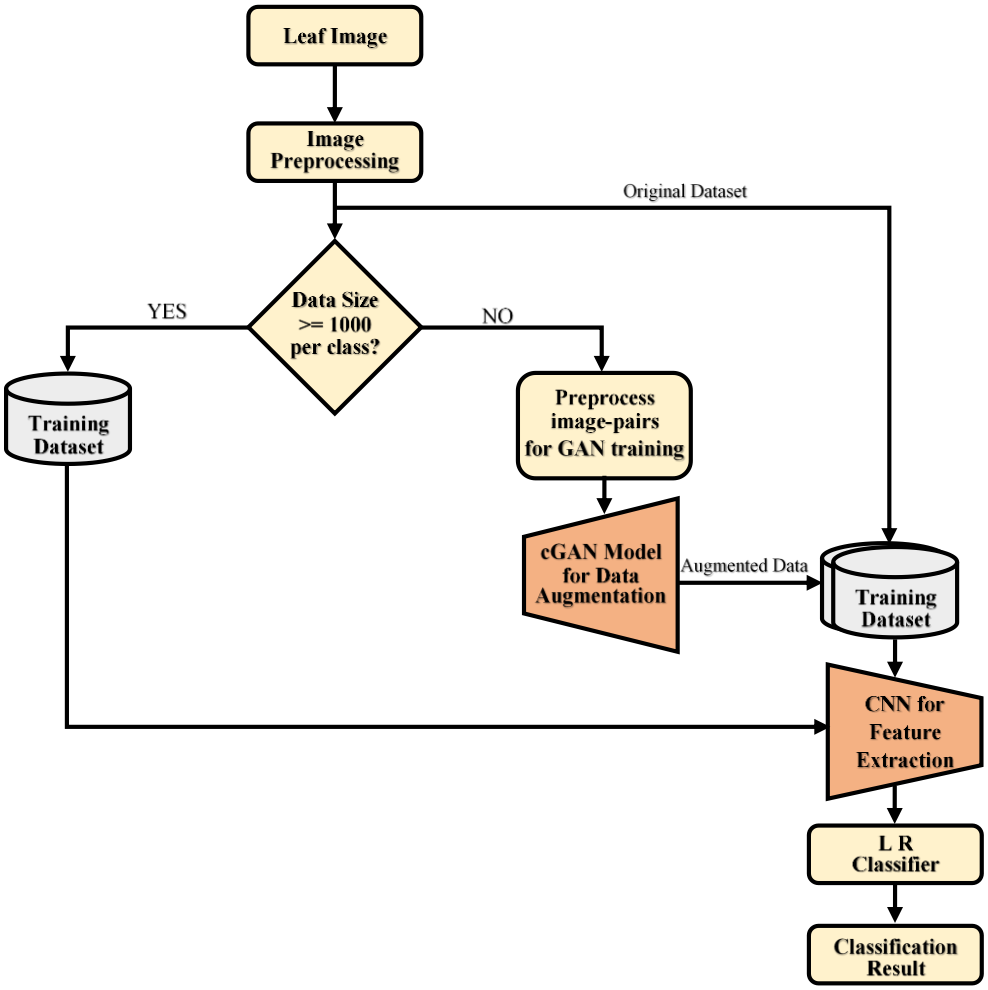

FIGURE 7. General flowchart of our proposed method showing the entire process from input to classification of the leaves.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

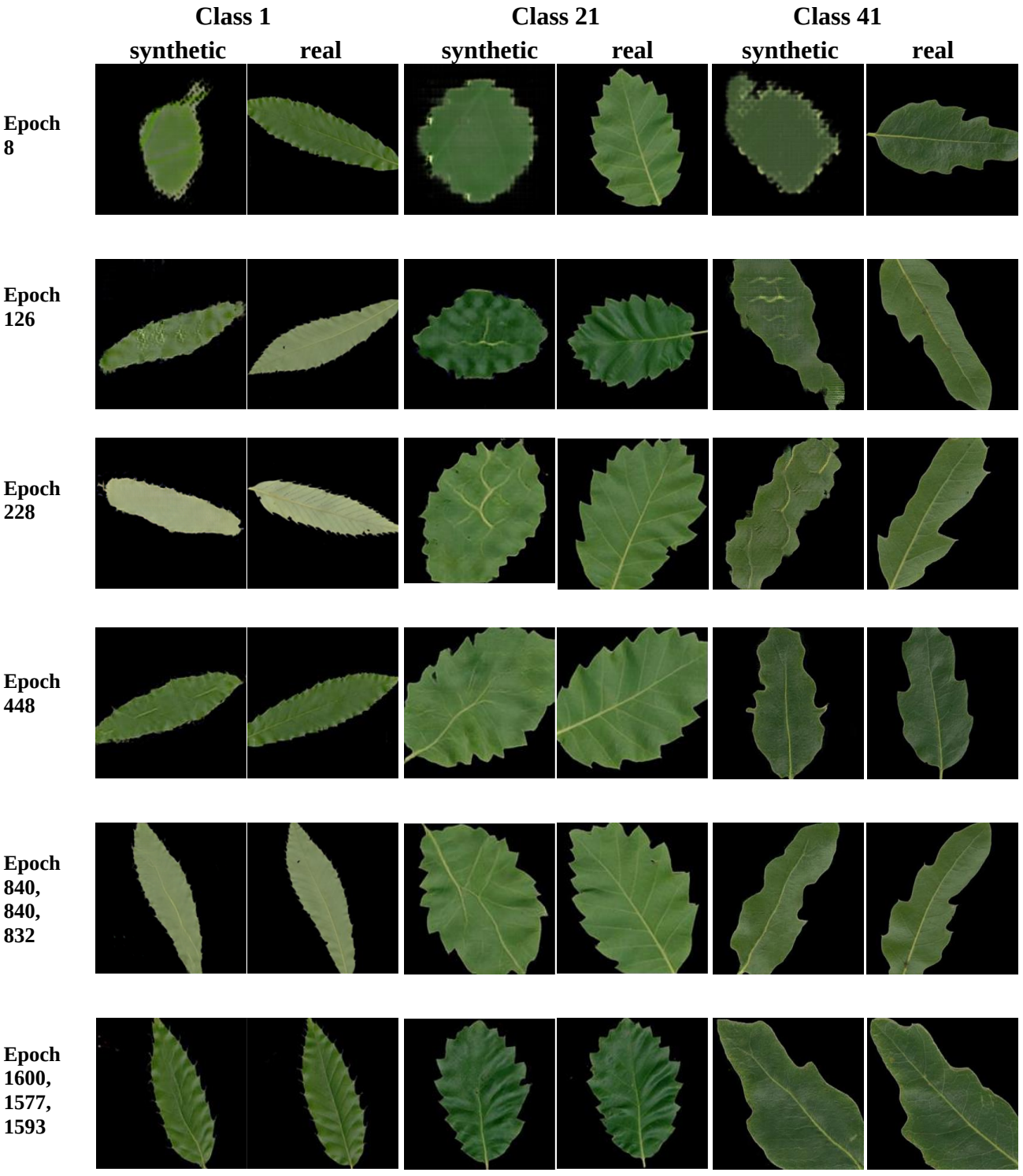

FIGURE 8. **Result of the Data Augmentation on the MalayaKew dataset using conditional Generative Adversarial Networks: It can be observed how the GAN model made progress from the beginning, finding it difficult to create what looks like a leaf image and getting its way through with increasing epochs.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

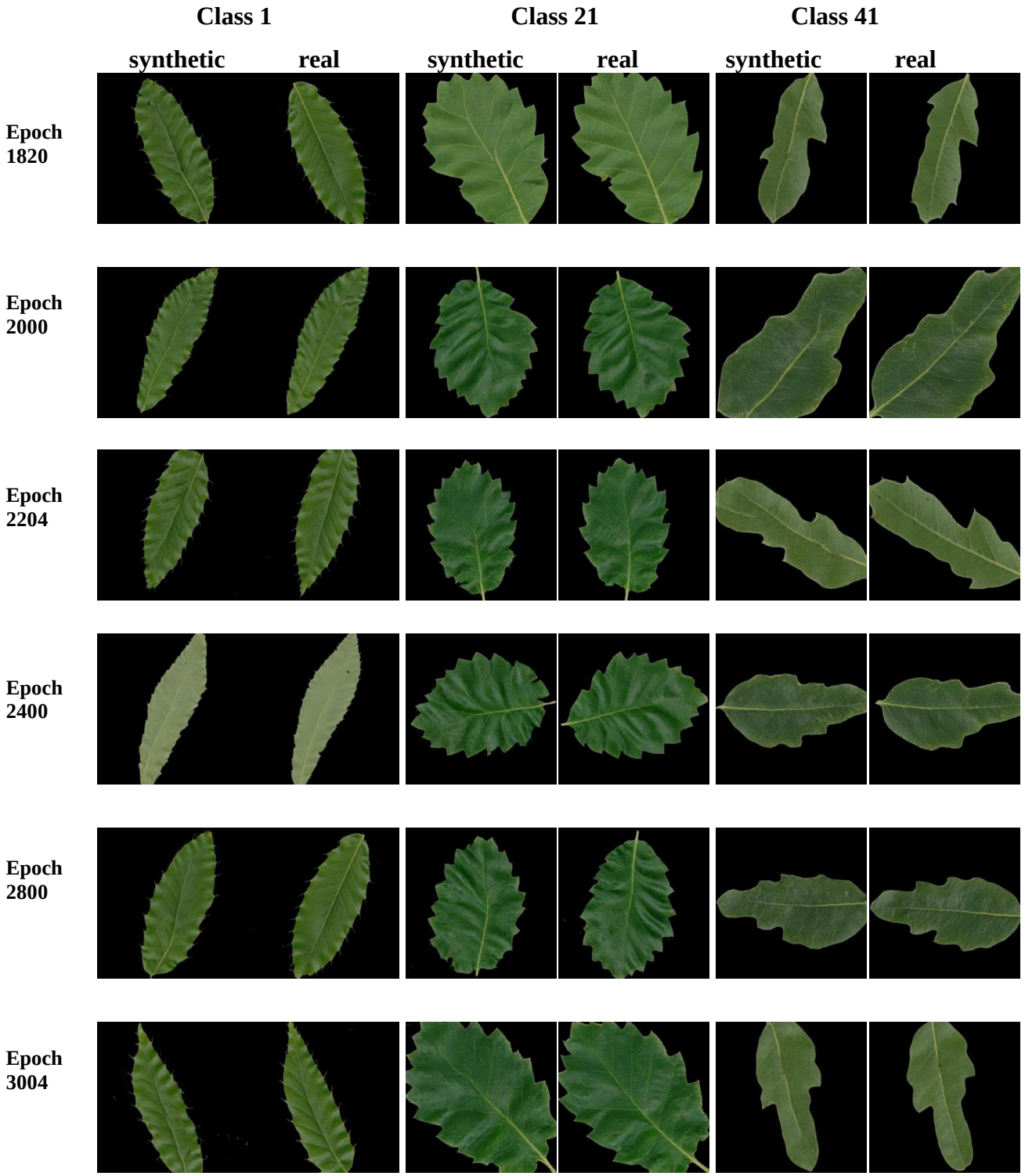

FIGURE 9. **Result of the Data Augmentation on the MalayaKew dataset using conditional Generative Adversarial Networks: It can be observed how the GAN model made progress from the beginning, finding it difficult to create what looks like a leaf image and getting its way through with increasing epochs.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

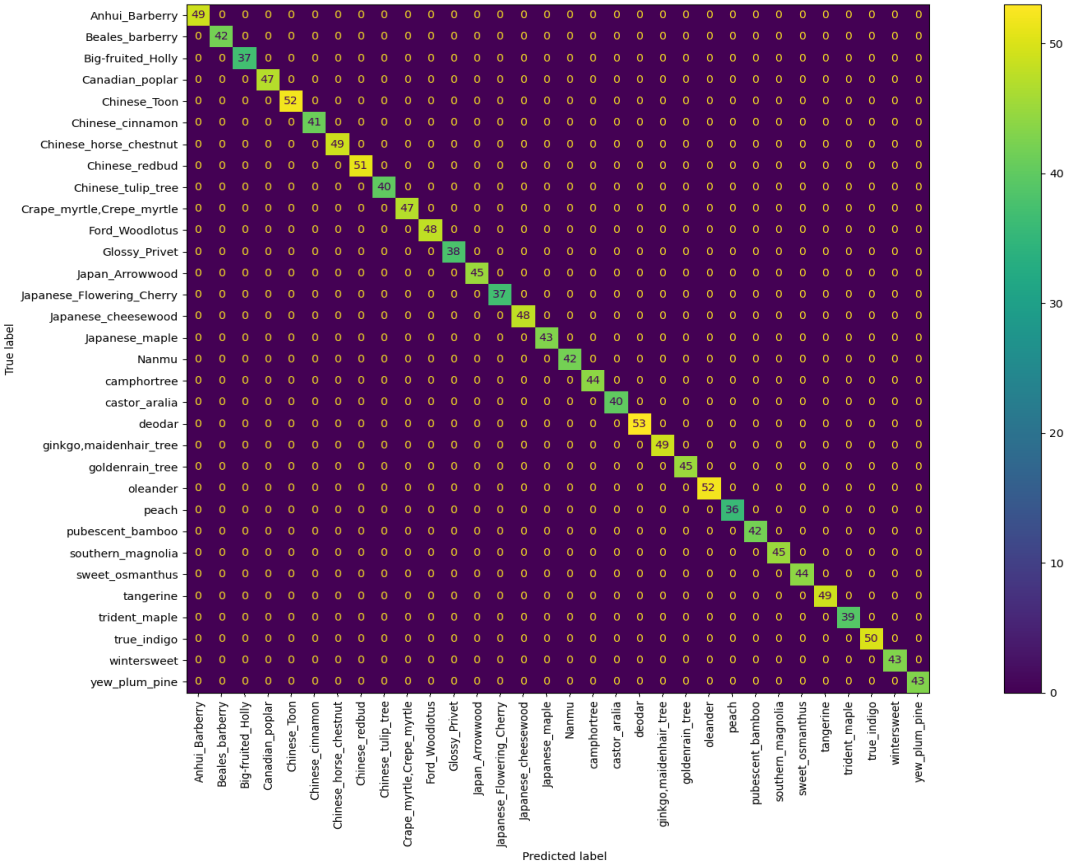

FIGURE 10. **Confusion Matrix of the result of the model on the flavia dataset.**

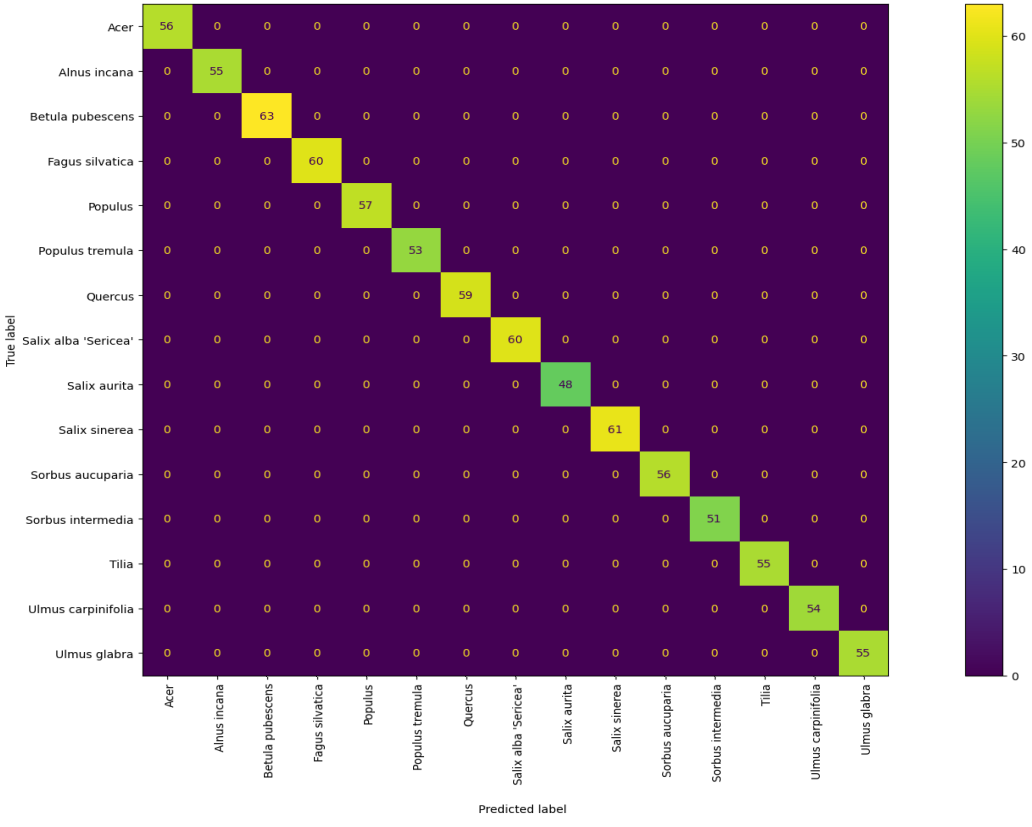

FIGURE 11. **Confusion Matrix of the result of the model on the Amazon Forest dataset.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

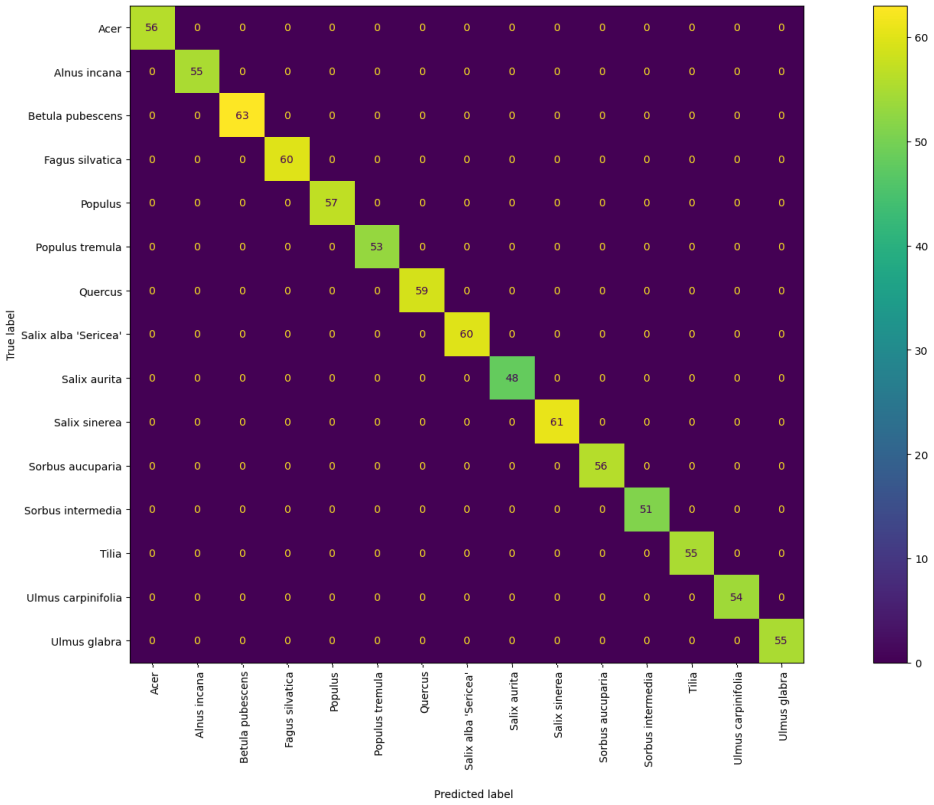

FIGURE 12. **Confusion Matrix of the result of the model on the Swedish dataset.**

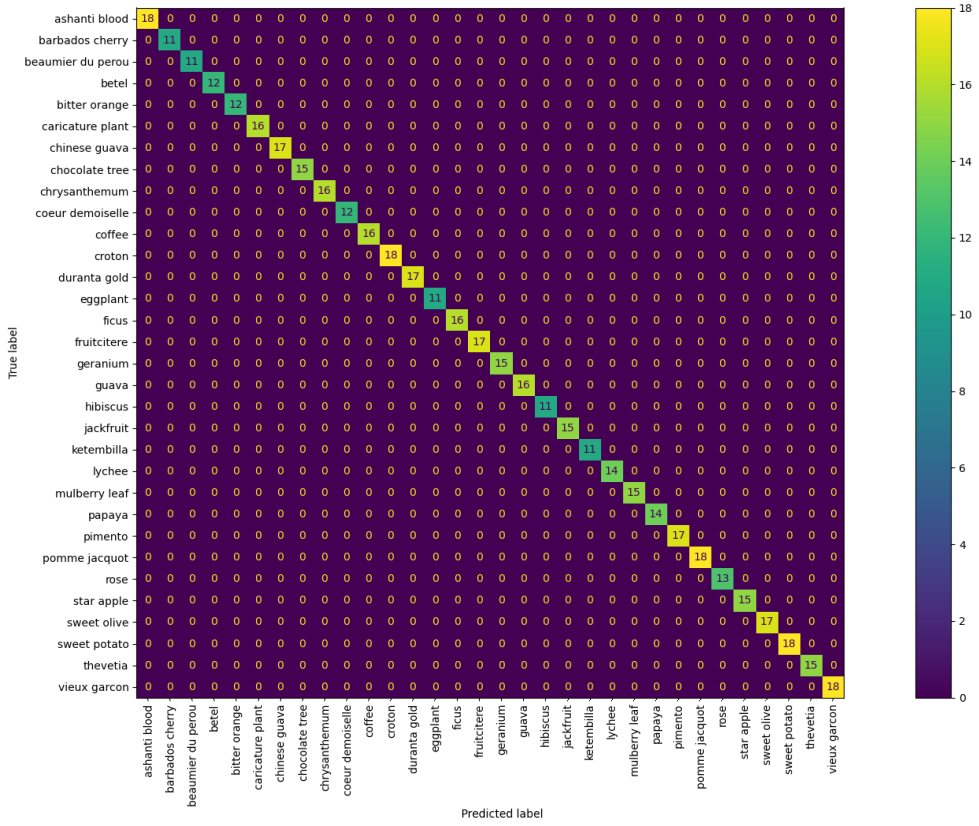

FIGURE 13. **Confusion Matrix of the result of the model on folio dataset.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

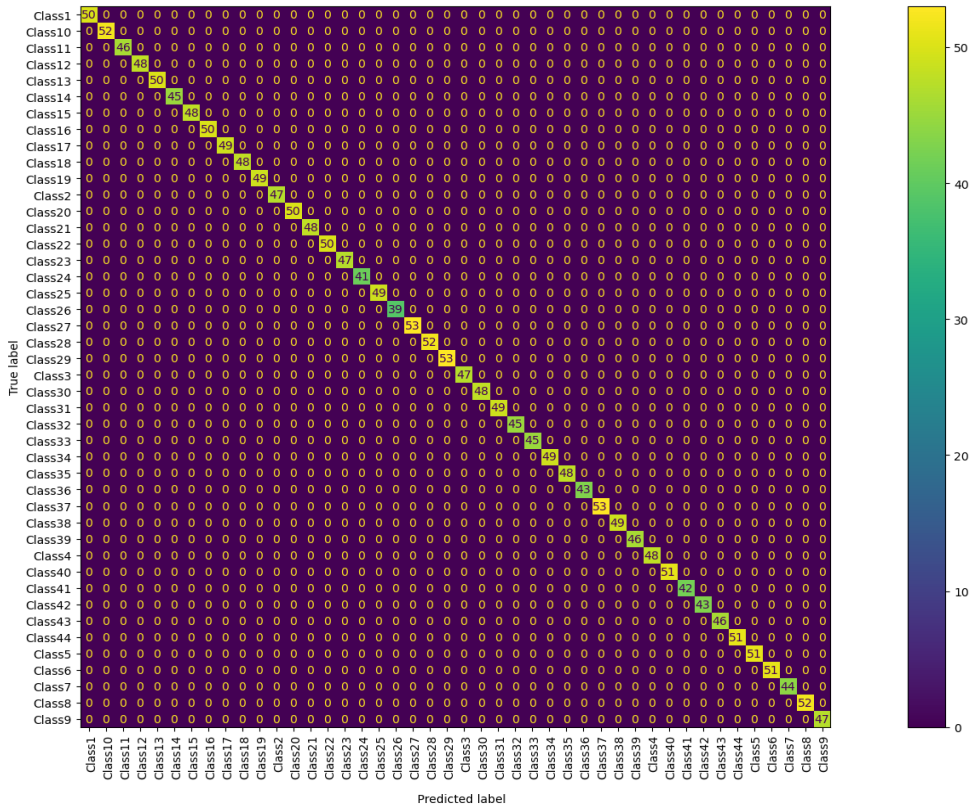

FIGURE 14. Confusion Matrix of the result of the model on MalayaKew_D1 dataset.

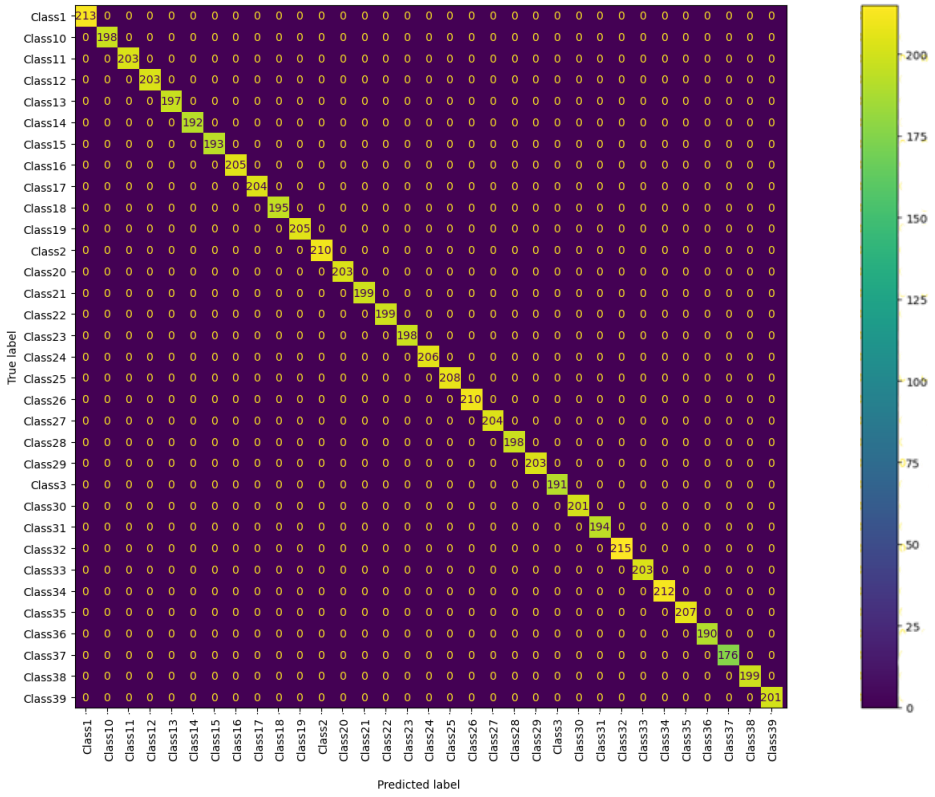

FIGURE 15. Confusion Matrix of the result of the model on both combination of MalayaKew-D1 and Synthetic dataset generated using conditional Generative Adversarial Networks.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

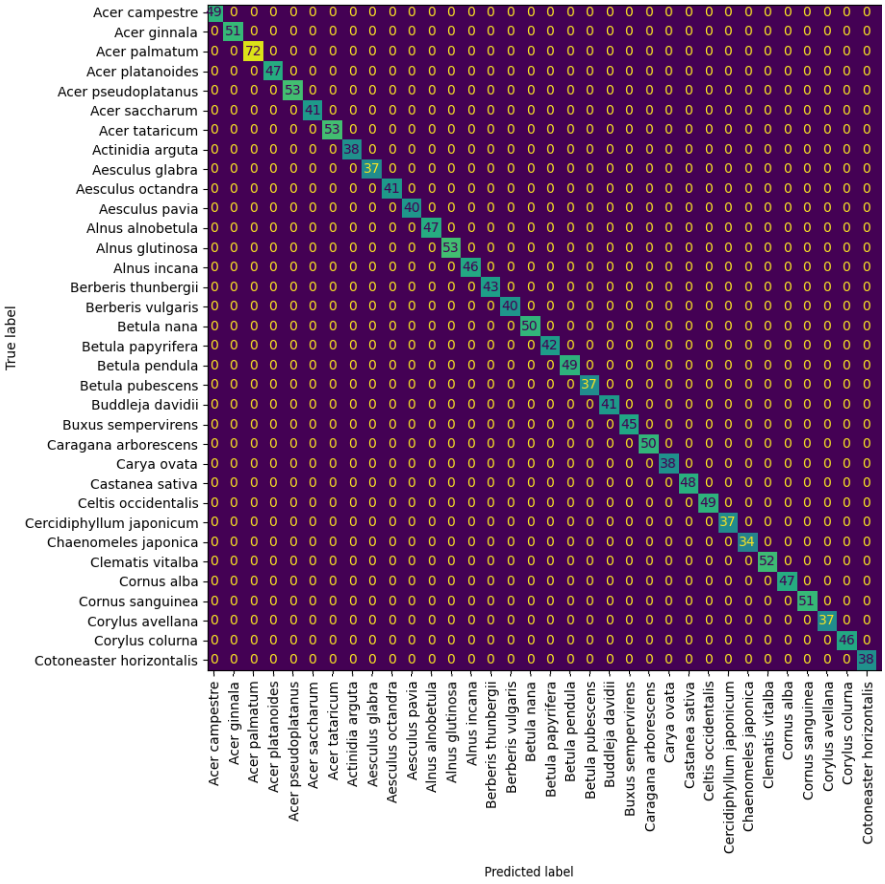

FIGURE 16. Confusion Matrix of the result of the model on the mew_2012 dataset.

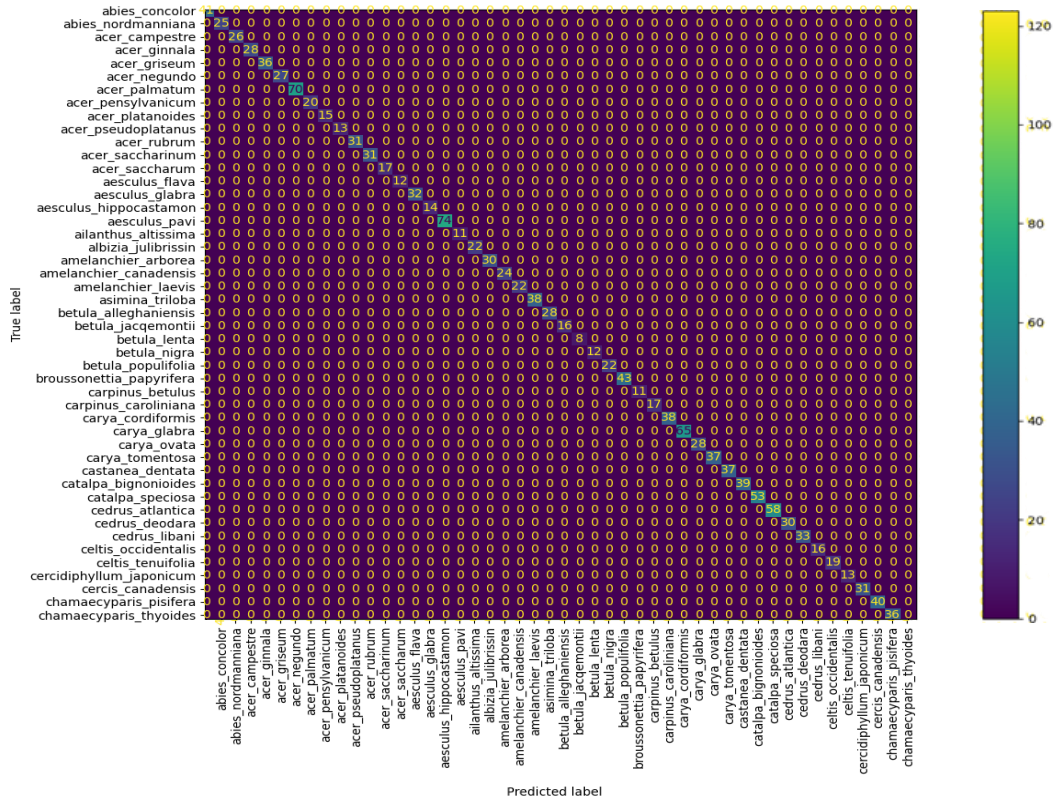

FIGURE 17. Confusion Matrix of the result of the model on the LeafSnap dataset.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

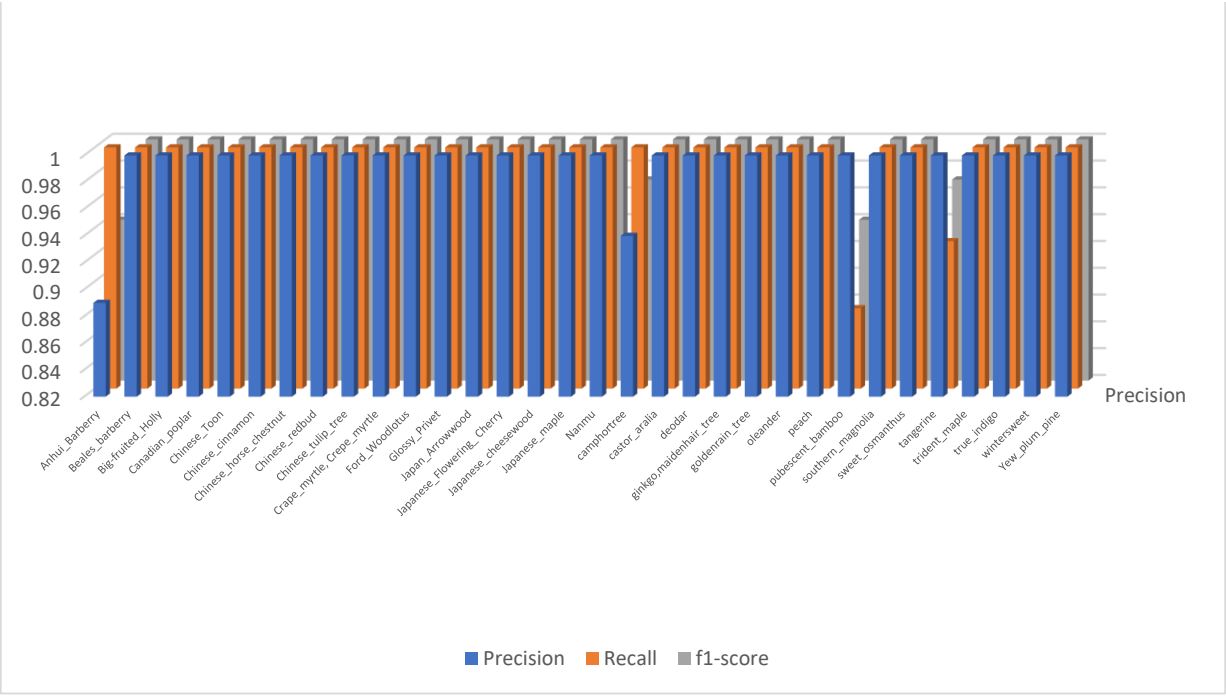

FIGURE 18. **Result of the model on the flavia dataset.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

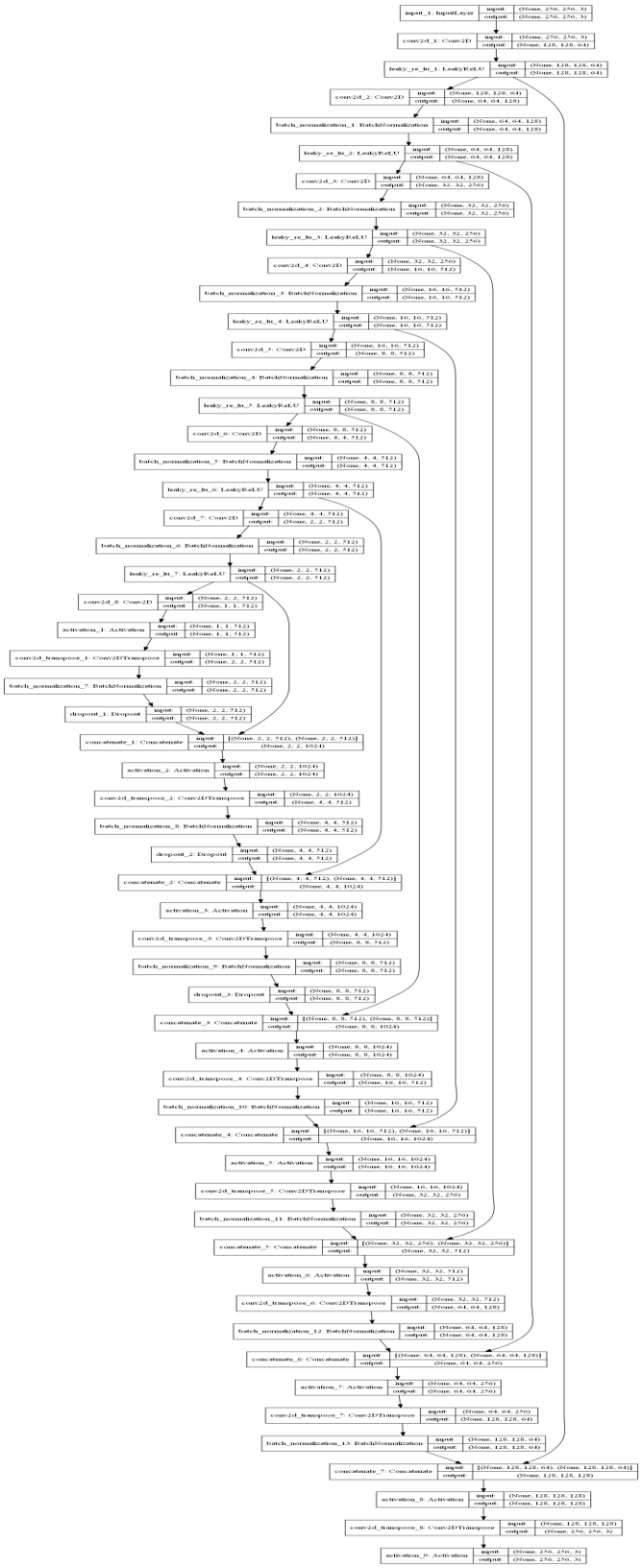

FIGURE 19. Generator Architecture of the cGAN model.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3131726, IEEE Access

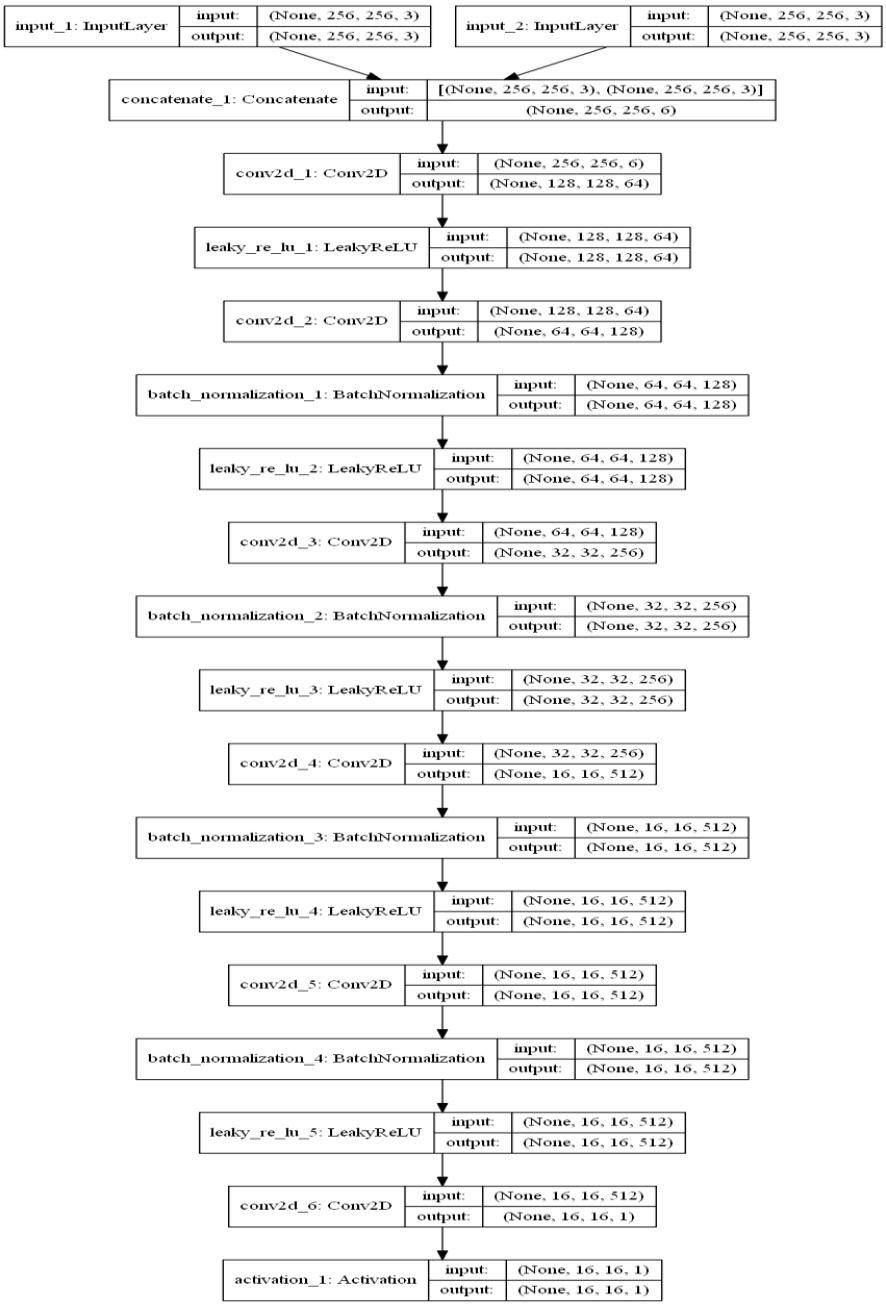

FIGURE 20. Discriminator Architecture of the cGAN model.