

07 Apr 2022

Plant Identification in a Combined-Imbalanced Leaf Dataset

Viraj K. Gajjar

Anand K. Nambisan

Kurt Louis Kosbar

Missouri University of Science and Technology, kosbar@mst.edu

Follow this and additional works at: https://scholarsmine.mst.edu/ele_comeng_facwork

 Part of the [Electrical and Computer Engineering Commons](#)

Recommended Citation

V. K. Gajjar et al., "Plant Identification in a Combined-Imbalanced Leaf Dataset," *IEEE Access*, vol. 10, pp. 37882 - 37891, Institute of Electrical and Electronics Engineers (IEEE), Apr 2022.

The definitive version is available at <https://doi.org/10.1109/ACCESS.2022.3165583>



This work is licensed under a [Creative Commons Attribution 4.0 License](#).

This Article - Journal is brought to you for free and open access by Scholars' Mine. It has been accepted for inclusion in Electrical and Computer Engineering Faculty Research & Creative Works by an authorized administrator of Scholars' Mine. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.

Received March 2, 2022, accepted March 30, 2022, date of publication April 7, 2022, date of current version April 13, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3165583

Plant Identification in a Combined-Imbalanced Leaf Dataset

VIRAJ K. GAJJAR¹, (Graduate Student Member, IEEE),
ANAND K. NAMBIAN¹, (Graduate Student Member, IEEE),
AND KURT L. KOSBAR¹, (Member, IEEE)

Electrical and Computer Engineering Department, Missouri University of Science and Technology, Rolla, MO 65409, USA

Corresponding author: Viraj K. Gajjar (vgf4c@mst.edu)

ABSTRACT Plant identification has applications in ethnopharmacology and agriculture. Since leaves are one of a distinguishable feature of a plant, they are routinely used for identification. Recent developments in deep learning have made it possible to accurately identify the majority of samples in five publicly available leaf datasets. However, each dataset captures the images in a highly controlled environment. This paper evaluates the performance of EfficientNet and several other convolutional neural network (CNN) architectures when applied to a combination of the LeafSnap, Middle European Woody Plants 2014, Flavia, Swedish, and Folio datasets. To normalize the impact of imbalance resulting from combining the original datasets, we used oversampling, undersampling, and transfer learning techniques to construct an end-to-end CNN classifier. We placed greater emphasis on metrics appropriate for a diverse-imbalanced dataset rather than stressing high performance on any one of the original datasets. A model from EfficientNet's family of CNN models achieved a highly accurate F-score of 0.9861 on the combined dataset.

INDEX TERMS Leaf dataset, imbalanced dataset, convolutional neural networks, transfer learning, plant identification.

I. INTRODUCTION

Plants with ethnopharmacological uses are a primary source of medicine. About 80% of 122 plant-derived drugs are related to their original ethnopharmacological purposes [1]. A fundamental step in conducting research on plant-derived compounds is establishing the plant's botanical identity. Medical research on compounds derived from plants benefits from an expertise in botany. Appropriate use of botanical nomenclature can help avoid errors and ambiguities in phytomedical, ethnopharmacological, and other research on plants [2]–[5].

Plant identification is the process of comparing features of a plant to previously collected and categorized specimens. This process helps connect the specimen with a particular species, which lets one recognize the specimen's properties. The method of plant identification is instrumental to plant taxonomy, and botanical nomenclature [2]–[5]. A faster, automated, and more reliable method of plant species identification would be helpful in phytochemical research. Plant

identification also has applications in site-specific weed management in plant agriculture, and plant phenotyping [6], [7].

Various plant organs can be used to identify a plant species, e.g., flowers, leaves, stem, fruit, or even the entire plant. Due to their distinctive features, leaves are particularly useful for identifying plants. More than 100 studies have used images of leaves to identify plants [8]. A leaf consists of a blade and a petiole. The blade consists of the following sub-parts: apex, margin, veins, midrib, and base, as shown in Fig. 1, that can be used as features for classification. Several computer vision techniques can use images of leaves to extract features that are potentially distinct amongst different species to identify plants [8].

Deep learning (DL) is a subset of machine learning methods. DL algorithms are inspired by biological neurons' structure and function. DL has recently experienced rapid growth due to increased data availability and substantial improvements in hardware technologies. DL uses multiple layers of information processing to build abstractions of data, from which features can be extracted and patterns classified. One of the widely used algorithms in DL is the convolutional neural network (CNN), which is extensively used for image classification [9]–[15]. A CNN is able to extract features in

The associate editor coordinating the review of this manuscript and approving it for publication was Charith Abhayaratne¹.

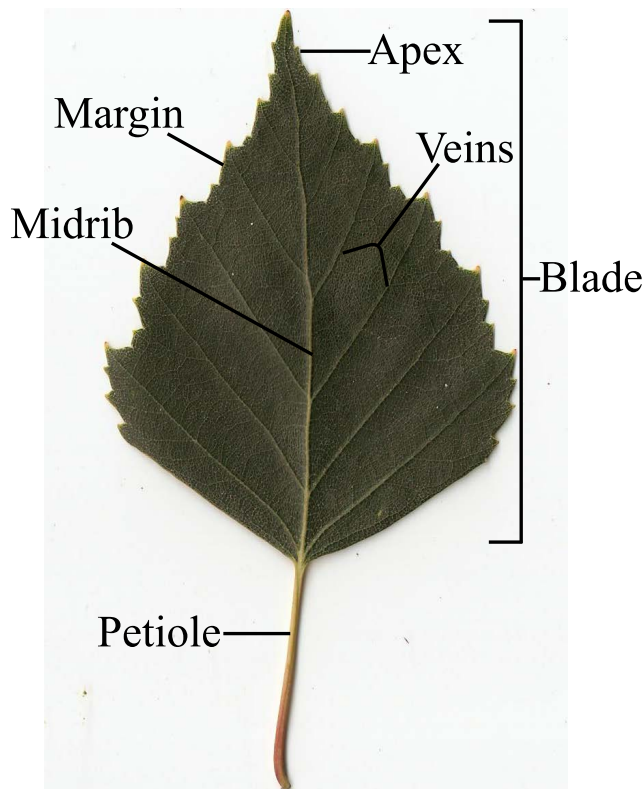


FIGURE 1. Parts of leaves that can be potentially used as features by computer vision algorithms for classification.

the form of filters from the leaf images and classify plants based on the features learned, given that the dataset is large enough. Numerous CNN architectures have been proposed to perform leaf classification tasks, and are tested on several publicly available leaf datasets. These approaches have successfully achieved high classification accuracies on the datasets on which they were tested [8], [16]–[19]. However, almost all of the previous work using DL has been done on individual leaf datasets, which does not factor in the impact of varying environments in which the images of a particular dataset are captured, and varying phenotypes of the same species over different regions. Many DL approaches to leaf identification use CNNs only as feature extractors and requires a separate classifier operation such as support vector machines or random forests. This work focuses on using a CNN to perform the entire classification process, rather than just focusing on feature extraction.

This paper proposes a new leaf dataset called F2LSM (Flavia, Folio, LeafSnap, Swedish, and MEW 2014). F2LSM dataset was created by combining the following five publicly available leaf datasets: LeafSnap [20], Middle European Woods (MEW) 2014 [21], Flavia [22], Swedish [23], and Folio [24]. The goal is to create a more comprehensive dataset with appropriate botanical nomenclature. Dataset pre-processing was performed prior to combining the datasets to remove classes with incomplete scientific names and classes that were judged to be poor image quality. Due to LeafSnap and MEW 2014 datasets' imbalanced nature and

certain overlapping species, while combining the datasets, a simple combination would result in a highly imbalanced dataset. This imbalance may potentially harm a CNN's performance by creating a bias towards the classes with more samples. To mitigate this problem, classes with a relatively large number of image samples, known as majority classes, are undersampled. Additionally, classes with a relatively small number of image samples, known as minority classes, are oversampled [25]–[27]. The resulting F2LSM dataset has 42420 leaf images belonging to 374 distinct classes of plant species arranged in folders named after their genus and species. The F2LSM dataset, along with the code, is publicly available at the following website: https://scholarsmine.mst.edu/research_data/8/.

One of DL's serious problems is that it depends on a massive amount of training data compared to other traditional ML methods. The initial layers of DL models generally extract more elementary features such as edge and texture, and the later layers build on top of it to extract more abstract features pertaining to the task at hand. Transfer learning (TL) leverages this property by relaxing the hypothesis that the training data must be independent and identically distributed (i.i.d) with the test data. Therefore, it is not required to train the target model from scratch, which can significantly reduce the amount of training data and time required to train a model in the target domain [28]. This paper applies the concept of TL by using pre-trained weights and biases of CNN architectures trained on the ImageNet dataset [29], and then fine-tuning them to classify the images in the F2LSM dataset. The CNN architectures from the EfficientNet family of models, B0 to B6, are used for the classification task [15], and their performance is compared with the following other architectures: VGG19 [12], InceptionV3 [30], ResNet50V2 [13], DenseNet121 [31], Xception [32], and MobileNetV2 [33]. Fig. 2 summarizes the process of combining the datasets and training a CNN. A significant number of approaches for classifying plants based on leaf images [34]–[38] have not considered the imbalanced nature of the datasets used, which may reduce their effectiveness. This paper attempts to evaluate the performance of these models by using metrics intended for imbalanced data.

The major contributions of the paper can be summarized as follows:

- Clean and combine five publicly available leaf datasets into one F2LSM dataset, and make the dataset publicly available.
- Implement undersampling and oversampling to overcome potential bias introduced because of the imbalance in data.
- Create an end-to-end CNN-based leaf classifier using various CNN architectures and TL and obtain comparable accuracy on the proposed dataset compared to the state-of-the-art accuracy obtained on the individual datasets.
- Use metrics that consider the imbalanced nature of the dataset to test and validate the performance EfficientNet,

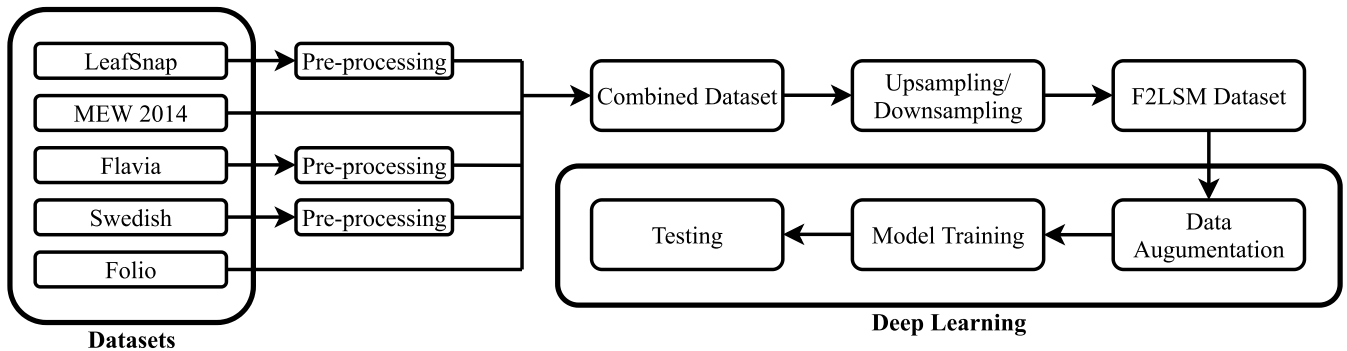


FIGURE 2. Flow Diagram of the procedure. Five publicly available datasets, LeafSnap, MEW 2014, Flavia, Swedish, and Folio, are combined into a single dataset after pre-processing. Resulting dataset is then balanced by upsampling the minority classes and downsampling the majority classes to form the F2LSM dataset. Finally, multiple CNNs are trained on the F2LSM dataset and their performance is assessed.

and compare its performance with other well-known CNN architectures.

The rest of the paper is structured in the following manner. Section II briefly summarizes relevant work in the field of plant identification. Cleaning and combining different datasets to create the F2LSM dataset is discussed in section III. Section IV discusses the CNN architectures used for training and testing the classifier. Section V discusses the experimental setup, the metrics used, and the results. Finally, Section VI provides conclusions and a summary of the results.

II. RELATED WORK

Plant species identification based on leaf images has been an active area of research [8], [16]–[18]. Kumar *et al.* [20] used features related to a leaf’s curvature to identify species in the proposed LeafSnap dataset and achieved a top-5 accuracy of 96.8%. Fourier descriptors for leaf contours combined with the nearest-neighbor classifier were used to classify leaf images in the proposed MEW 2012 dataset with an accuracy of 88.91% [21]. Wu *et al.* [22] proposed the Flavia dataset with 32 different species of plants and 12 leaf features in combination with a probabilistic neural network (PNN) to achieve a classification accuracy of 90%. Features based on the geometry, eigenleaves, and grey-level co-occurrence matrix (GLCM) were used to train a support vector machine (SVM) classifier for leaf identification by [39]. Munisami *et al.* [24] used shape features and color histogram with k-nearest neighbors to classify plant leaves in the Folio dataset with an accuracy of 87.3%. Chaudhry *et al.* [34] used the concept of contour-based 2D shape matching to identify plant species from occluded leaf images by using the contour information extracted from publicly available leaf datasets. Kumar *et al.* [35] used morphological features extracted using a multilayer perceptron with adaboosting to train a classifier and attained an accuracy rate of 95.42% on the Flavia dataset.

Some methods use pre-trained CNN models to extract features from leaf images and then use those features to train ML classifiers [36], [40]. Wang *et al.* [37] proposed a

novel counting-based leaf recognition method that effectively combines all of the three significant characteristics – leaf contour, texture, and vein – in leaf images by using elliptical half Gabor, which is then convolved with grayscale leaf images to calculate histogram of line patterns used a descriptor to train SVM. This method achieved a classification accuracy of 98.40% on the Swedish dataset and 97.83% accuracy on the Flavia dataset. Song *et al.* [38] proposed a novel attention branch-based convolutional neural network (ABCNN) to identify plant species with highly similar leaves, and was able to attain a classification accuracy of 91.43% on a special dataset created from LeafSnap. Raj *et al.* [19] proposed a dual deep learning architecture (DDL) that combines MobileNet and DenseNet-121 architectures to extract features and then passes those features to ML classifiers and fully connected layers (FCL). The DDLA method [19] achieved the highest accuracy of 98.71%, 96.38%, and 99.41% on the Flavia, Folio, and Swedish datasets, respectively.

III. F2LSM DATASET

A. CLEANING AND COMBINING DATASETS

The F2LSM dataset was created by combining five publicly available datasets: LeafSnap [20], MEW 2014 [21], Flavia [22], Swedish [23], and Folio [24]. Table. 1 provides a links to a webpages where each dataset can be found, along with information about number of classes and number of images samples in each dataset, and a representative image from each dataset.

1) LeafSnap DATASET

The LeafSnap dataset was first introduced as a part of a mobile application called LeafSnap, which helps users identify plants from photographs of their leaves [20]. The species in the LeafSnap dataset covers plants found in the North Eastern United States. The dataset consists of two categories of images: field images (low-quality images created by mobile devices in outdoor environments) and lab images (high-quality images of pressed leaves from the Smithsonian collection). This dataset is available at the following website: <http://leafsnap.com/dataset/>.

TABLE 1. Dataset’s websites along with information about number of classes, number of image samples, and an example of an image sample from each dataset.






Dataset	Dataset’s URL	Number of Classes	Number of Image Samples	Representative Image Sample
LeafSnap	https://bit.ly/37fLgJP	185	30866	
MEW 2014	https://bit.ly/3NDCC8u	201	15074	
Flavia	https://bit.ly/3x1O6gn	32	1907	
Swedish	https://bit.ly/35zuULo	15	1125	
Folio	https://bit.ly/3uMtFBc	32	637	



FIGURE 3. Class-specific cropping in the LeafSnap dataset.

The lab images had calibration marks on the right and bottom edges of the images, which were not present in the field images and in other datasets. Since partial calibration information of this nature might not be very helpful when the datasets are combined, it was removed from all the lab images, as shown in Fig. 3. After cropping lab images, we put the field and lab images in the same directory for each respective class. The LeafSnap dataset had 27 classes, most belonging to the genus *pinus*, along with some other species with low-quality images. These images contained a minimal amount of information that an ML algorithm can use, and showed no difference amongst different species, Fig. 4. We removed these 27 classes from the LeafSnap dataset.

2) MEW 2014 DATASET

The MEW 2012 dataset was first introduced in [21] for the recognition of woody species in Central Europe. The dataset consisted of leaf samples from 151 unique species collected from the Czech Republic. Since then, the dataset has been expanded to incorporate 50 more species and is now called MEW 2014. This dataset is available at the following website upon request: <http://zoi.utia.cas.cz/node/662>. No modifications were made to this dataset when combining, before combining it with other datasets.

3) FLAVIA DATASET

The Flavia dataset was first introduced in [22] due to a lack of a standard plant leaf dataset and to create a classification



FIGURE 4. Image samples of different classes removed from the LeafSnap dataset, each image belongs to a different class.

benchmark. This dataset is available at the following website: <http://flavia.sourceforge.net/>. The Flavia dataset had all its images in a common directory, so we separated them into directories named after their respective botanical name.

4) SWEDISH DATASET

The Swedish leaf dataset was first introduced in [23] to create a computer vision algorithm for leaf identification for the Swedish Museum of Natural History. The dataset is available at the following website: <https://www.cvl.isy.liu.se/en/research/datasets/swedish-leaf/>. Before combining the datasets, we removed four classes from the Swedish dataset due to a lack of a proper scientific name.

5) FOLIO DATASET

Folio dataset has pictures taken from the farm of the University of Mauritius [24]. The dataset consists of 20 image samples for each of the 32 species represented. This dataset is available on the UCI Machine Learning Repository website at the following URL: <https://archive.ics.uci.edu/ml/datasets/Folio#>. This dataset was left unaltered before combining it with other datasets.

We combined these five datasets into one dataset with 374 classes, naming it the F2LSM dataset. The size of images was primarily unaltered when combining the dataset, except for when cropping was performed in LeafSnap dataset. Table 2 summarizes the range of image sizes in each dataset.

B. OVERSAMPLING AND UNDERSAMPLING

The combined dataset was highly imbalanced due to the imbalanced nature of LeafSnap and MEW 2014 datasets, overlapping classes, and some classes with a minimal number of images. Some classes had fewer than ten images in the

TABLE 2. Range of image sizes in different datasets. All numbers indicate number of pixels.

Dataset	Minimum width	Maximum width	Minimum height	Maximum height	Smallest image/s	Largest image/s
Swedish	364	2550	652	4125	465 × 652	2412 × 4125
Folio	1141	4160	1726	4160	1152 × 1726	4160 × 3120
Flavia	1600	1600	1200	1200	1600 × 1200	1600 × 1200
MEW 2014	162	3504	115	4956	309 × 133	3504 × 4956
LeafSnap	210	800	194	800	512 × 194	800 × 800
F2LSM	162	4160	115	4956	309 × 133	3504 × 4956

combined dataset, while others had more than 300. The class distribution of the dataset was adjusted by randomly oversampling the minority classes and undersampling the majority classes. This method helps select more samples from one underrepresented class and creates a bias to select more from that class than others. One of the basic ways to oversample a class is to randomly select samples from the desired class and create copies of the data [25].

Instead of simply creating copies of the image samples in minority classes, we use the following method to implement oversampling. If the minority class has N images, then each of those images are each put through the following three transformations to create $3N$ new images:

1) GAUSSIAN BLUR

A Gaussian blur is implemented by convolving an image with a 2D Gaussian kernel. The values of the kernel are from the Gaussian function (1), where x and y denote the position of the pixel and σ is a parameter that allows for adjustment of the width of the blur. Each pixel's new value would be the weighted average of pixel's old value and values of its neighboring pixels. The center pixel in the convolution operation would receive the highest weight from the Gaussian kernel, and the neighboring pixels will receive smaller weights depending on how far they are from the center. This operation blurs an image by acting as a low pass filter.

$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (1)$$

2) UNSHARP MASKING

Unsharp masking is a linear image processing technique that enhances the edges of an image. It is implemented by low pass filtering an image using a Gaussian blur and comparing it to the original image. This difference is then scaled and added to the original image. This operation is summarized in (2), where x and y denote the position of the pixel, $f_{sharp}(x, y)$ is the unsharp masked image, $f(x, y)$ is the original image, $f_{smooth}(x, y)$ is the low pass filtered image, and k is the scaling constant.

$$f_{sharp}(x, y) = f(x, y) + k(f(x, y) - f_{smooth}(x, y)) \quad (2)$$

3) RANK FILTER

A rank filter of rank- k is a non-linear operation that sorts the pixel values in a kernel of M pixels and assigns the k^{th} value to the center point in the kernel. A rank filter with



FIGURE 5. Different image transforms applied for oversampling classes with image samples less than 10. From left to right is the original image, a Gaussian blurred image, a rank filtered image, and an unsharp masked image.

$k = 1$ would be the min filter, $k = M$ would be the max filter, and $k = \frac{M+1}{2}$ would be the median filter. Random rank values between 1 and 9 were chosen for a 3-by-3 kernel to implement rank filtering.

Fig. 5 demonstrates the effect of implementing these transforms. Undersampling can be implemented by randomly removing image samples from the majority classes [26]. Since undersampling can potentially discard useful samples, we only applied it to classes with more than 300 images. Fig. 6 shows the datasets' histogram before and after implementing oversampling and undersampling. The inset in Fig. 6 shows that before sampling, there were nine classes with number of image samples close to ten, and after sampling, there is no class with number of image samples less than or equal to ten.

IV. DEEP TRANSFER LEARNING

A. CONVOLUTIONAL NEURAL NETWORKS

Traditional computer vision algorithms require the designer to indicate which key features need to be extracted and how they can be extracted, whereas a CNN does not require humans to identify features in the image. A CNN architecture consists of multiple layers, including an input layer, an output layer, convolutional layers, rectified linear unit (ReLU) layers, pooling layers, and fully connected layers, as shown in Fig. 7. When processing the information in the forward direction, the convolutional layers convolve multiple filters with the input volume to generate an activation map for each filter. The convolution operation is followed by some non-linear activation function to increase non-linear properties of the network e.g. ReLU: $f(x) = \max(0, x)$ [9]. The pooling layer reduces the number of parameters in the network by non-linearly downsampling the activation maps or an image,

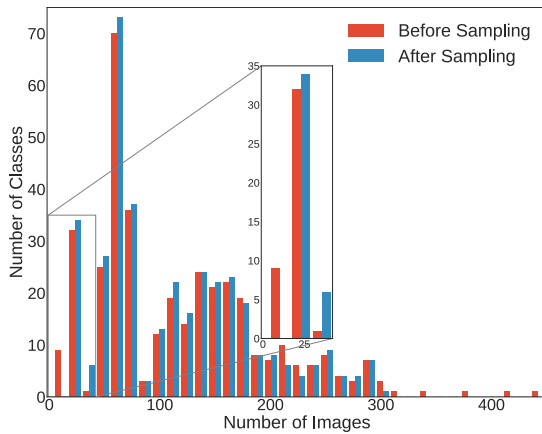


FIGURE 6. Histogram with 35 bins of the number of image samples per class before and after implementing oversampling and undersampling. The inset demonstrates that there were nine classes with less than ten images before sampling, and after sampling there are zero classes with less than ten image samples.

thus reducing the number of computations in the network. After multiple convolutional and pooling layers, a classification decision is made by fully connected layers. Once the feed-forward part is done, the CNN optimizes each layer's filters through a backpropagation mechanism. These filters, once optimized, extract important features representing the input image.

B. TRANSFER LEARNING

The combined dataset is imbalanced, has a large number of classes, and some of the classes have insufficient data required to train a CNN. We use transfer learning to initialize the CNN to help overcome the problem of insufficient and imbalanced data to a certain extent, as it relaxes the hypothesis that the training data must be i.i.d. with the test data. Therefore, the knowledge gathered while training a CNN on a diverse dataset like ImageNet [29] can be used while training a CNN for other classification tasks. To implement TL, the most commonly used method is to remove the last fully connected layer of the pre-trained CNN and replace it with a fully connected layer appropriate for the new number of classes. In this paper different CNN architectures—EfficientNet [15] architectures B0 to B6, VGG19 [12], InceptionV3 [30], ResNet50V2 [13], DenseNet121 [31], Xception [32], and MobileNetV2 [33]—trained on the ImageNet dataset are used for TL. We fine-tune the last few layers of the trained CNN architectures to classify 374 classes of leaves, as shown in Fig. 8. This technique helps significantly reduce the training time and achieve higher accuracies, including on imbalanced datasets. Table 5 in the Experiments section demonstrates this improvement.

V. EXPERIMENTS, RESULTS, AND ANALYSIS

A. STRATIFIED K-FOLD CROSS VALIDATION

In order to perform robust validation, we implemented stratified 5-fold cross-validation. Firstly, 10% of the dataset was set aside as a hold-out test set. We used stratification to ensure

TABLE 3. Image augmentation types with their respective ranges.

Augmentation type	Range
Random flip	horizontal and vertical
Random rotation	$[-0.5\pi, 0.5\pi]$
Random contrast	$[0.7, 1.3]$
Random zoom	$[0.1, 0.3]$
Random translation	$[0.2, 0.2]$

that the test set had a population that best represented the population of the entire dataset. Then the remaining 90% of the dataset was split into five equal folds by using stratified sampling to avoid sampling bias [41]. Finally, we use four folds for training and the fifth fold for validation to train and validate the networks. We repeated this process five times, using each fold for validation once. Input image size was altered based on the recommended image size for each respective model. Before training the models, data augmentations were performed on the dataset to regularize and prevent overfitting. Table 3 summarizes types of augmentations and their respective ranges.

We trained added layers added to models for transfer learning until the validation loss did not improve for ten epochs. Once the validation loss plateaued, we unfroze the last few layers and fine-tuned them until the validation loss plateaued again, Fig. 9. It can also be noted that training loss leads to the validation loss before fine-tuning begins and then rapidly drops below the validation loss after fine-tuning starts. This is due to the high regularization implemented in the added layers.

B. METRICS

The combined dataset is imbalanced, as shown in Fig. 6 and hence a commonly used metric like accuracy may not be the best indicator of overall classification performance. This is a commonly used metric for the current task in most approaches mentioned in [section II]. A brief description of the metrics used and the method used to calculate them are listed below.

Top-k accuracy is one of the metrics used to evaluate model performance. It is the number of times the correct class has occurred in the top k predicted classes based on their probability scores. We also use macro-averaged variants of precision (3) and recall (5) to assess overall model performance across all classes. Macro-averaged precision is the average of the precisions of each class taken individually as described in 4. The individual class precision for class l can be calculated as follows:

$$Prec_l = \frac{TP_l}{TP_l + FP_l}, \quad (3)$$

where TP is the number of True Positives and FP is the number of False Positives. Then the macro-averaged precision can then be calculated as

$$Prec_{macro} = (1/N_c) \sum_{l=1}^{N_c} Prec_l. \quad (4)$$

where N_c is the total number of classes.

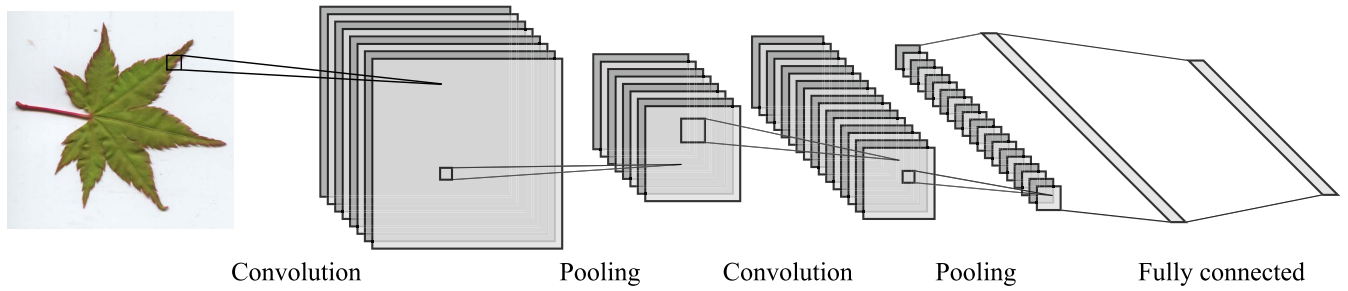


FIGURE 7. A typical CNN Architecture with multiple convolutional and pooling layers followed by fully connected layers for multi-class classification.

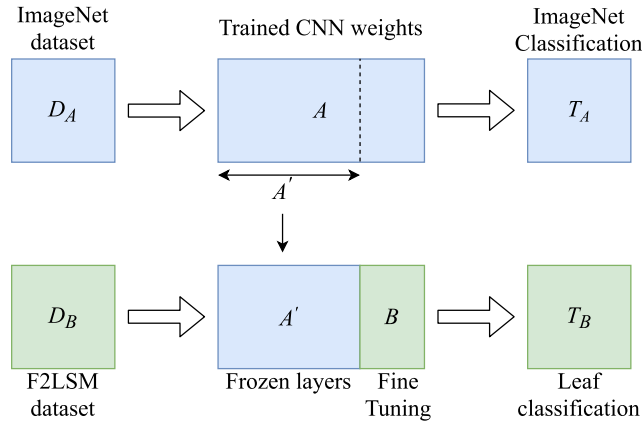


FIGURE 8. Transfer learning implemented on the F2LSM dataset. A' parameters learned for task A (T_A) are transferred to task B (T_B), and B parameters are fine-tuned.

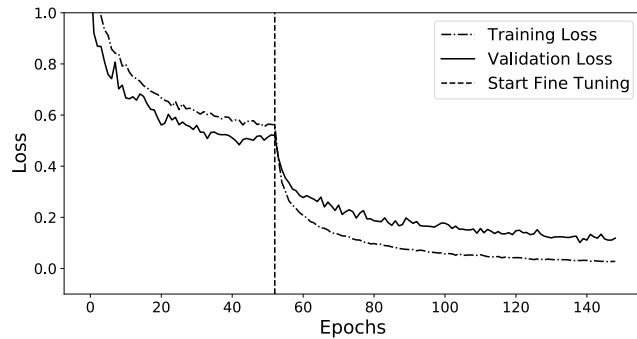


FIGURE 9. Loss trend while training EfficientNet B0, before and after fine-tuning the last few layers of the model.

In a similar fashion the macro-averaged recall is calculated as well as described in 6.

$$Rec_l = \frac{TP_l}{TP_l + FN_l} \quad (5)$$

$$Rec_{macro} = (1/N_c) \sum_{l=1}^{N_c} Rec_l, \quad (6)$$

where FN is the number of False Negatives.

Finally we also calculate the macro-averaged F-score. This is the average of the individual class F-scores and is calculated using 7.

$$F_{macro} = (1/N_c) \sum_{l=1}^{N_c} \frac{2 \times Prec_l \times Rec_l}{Prec_l + Rec_l}. \quad (7)$$

TABLE 4. Comparison of experimental results before and after performing oversampling and undersampling on the dataset.

Network	Before Balancing		After Balancing	
	Top-1 Accuracy	F-score	Top-1 Accuracy	F-score
MobileNetV2 [33]	0.9190	0.9400	0.9327	0.9416
DenseNet121 [31]	0.9455	0.9586	0.9618	0.9663
Xceptoin [32]	0.9475	0.9657	0.9731	0.9753
Inception V3 [30]	0.9599	0.9642	0.9696	0.9719
ResNet50V2 [13]	0.9377	0.9537	0.9591	0.9624
VGG19 [12]	0.9312	0.9388	0.9458	0.9495
EfficientNet B0 [15]	0.9335	0.9601	0.9590	0.9639
EfficientNet B1 [15]	0.9433	0.9618	0.9646	0.9697
EfficientNet B2 [15]	0.9462	0.9651	0.9683	0.9727
EfficientNet B3 [15]	0.9480	0.9681	0.9751	0.9778
EfficientNet B4 [15]	0.9506	0.9698	0.9771	0.9787
EfficientNet B5 [15]	0.9540	0.9728	0.9796	0.9811
EfficientNet B6 [15]	0.9593	0.9759	0.9841	0.9861

All the metrics were calculated for each fold on the hold-out test set, and the results were reported as $\mu \pm \sigma$ (mean \pm standard deviation).

C. RESULTS AND DISCUSSION

This section provides the results obtained while testing the trained networks before and after performing oversampling and undersampling.

1) EFFECT OF OVERSAMPLING AND UNDERSAMPLING

Table 4 shows the effect of oversampling and undersampling discussed in Section III in terms of the arithmetic means Top-1 accuracy and F-score obtained from stratified 5-fold cross-validation. Comparisons between all trained models show that balancing the dataset improves both Top-1 accuracy and F-score; however, the improvement is much more significant in the Top-1 accuracy metric than the F-score. For example, for EfficientNet B6, Top-1 accuracy improved by 1.66% after oversampling and undersampling, whereas F-score improved only by 0.2%. This disparity demonstrates that the F-score is a better metric when dealing with this imbalanced dataset.

2) PERFORMANCE ANALYSIS

The models used in this paper were trained on a Dell Precision 7920 Tower with an 8-core Intel Xeon Silver 4208 CPU @ 2.10 GHz, 64 GiB DIMM DDR4, and Nvidia Quadro RTX 4000 GPU. Table 5 summarizes the performance of different models on the hold-out test set. It can be observed that EfficientNet B6 achieves the highest Top-1 accuracy and F-score among other models. This superior

TABLE 5. Performance of different CNN architectures on the F2LSM dataset.

Network	Parameters	Input image size	Top-1 Accuracy	Precision	Recall	F-score
MobileNetV2 [33]	3.5M	224×224	0.9327±0.0011	0.9445±0.0013	0.9387±0.0017	0.9416±0.0018
DenseNet121 [31]	8.1M	224×224	0.9618±0.0018	0.9657±0.0018	0.9669±0.0011	0.9663±0.0015
Xception [32]	22.9M	299×299	0.9731±0.0015	0.9763±0.0012	0.9743±0.0009	0.9753±0.0017
Inception V3 [30]	23.9M	299×299	0.9696±0.0009	0.9723±0.0014	0.9715±0.0011	0.9719±0.0020
ResNet50V2 [13]	25.6M	224×224	0.9591±0.0012	0.9636±0.0011	0.9612±0.0016	0.9624±0.0020
VGG19 [12]	143.7M	224×224	0.9458±0.0012	0.9479±0.0011	0.9511±0.0018	0.9495±0.0014
EfficientNet B0 [15]	5.3M	224×224	0.9590±0.0007	0.9659±0.0012	0.9621±0.0013	0.9639±0.0011
EfficientNet B1 [15]	7.9M	240×240	0.9646±0.0018	0.9716±0.0019	0.9678±0.0021	0.9697±0.0017
EfficientNet B2 [15]	9.2M	260×260	0.9683±0.0011	0.9749±0.0020	0.9707±0.0013	0.9727±0.0016
EfficientNet B3 [15]	12.3M	300×300	0.9751±0.0020	0.9790±0.0013	0.9767±0.0014	0.9778±0.0010
EfficientNet B4 [15]	19.5M	380×380	0.9771±0.0011	0.9796±0.0015	0.9780±0.0013	0.9787±0.0019
EfficientNet B5 [15]	30.6M	456×456	0.9796±0.0007	0.9817±0.0008	0.9806±0.0011	0.9811±0.0011
EfficientNet B6 [15]	43.3M	528×528	0.9841±0.0014	0.9863±0.0009	0.9859±0.0010	0.9861±0.0009

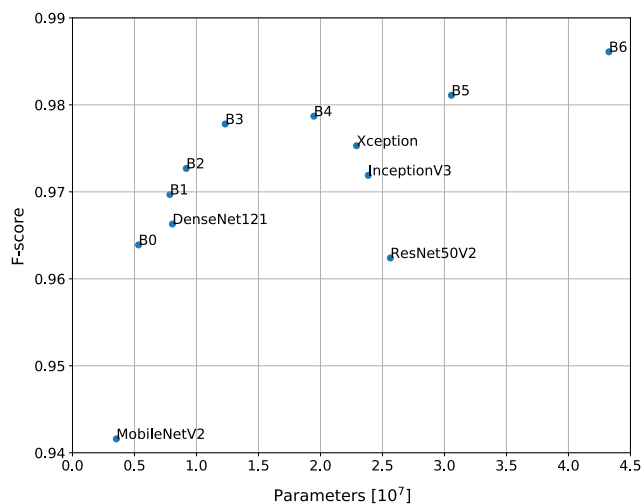


FIGURE 10. Number of model parameters vs. mean F-score. The plot shows that EfficientNet family of models perform better than other models for leaf identification in F2LSM dataset.

performance could be because EfficientNet B6 has 43.3M parameters, the largest input image size, and the fact that EfficientNet models are carefully balanced in terms of height, width, and resolution of the networks, which can lead to a better performance [15]. EfficientNet B6 took approximately 600 hours to train, which was roughly five times more when compared to the training time required for relatively smaller models such as MobileNet V2 and EfficientNet B0.

Even when considering models with similar input image sizes and a similar number of parameters, EfficientNet models generally perform better than other models on the F2LSM dataset. For example, EfficientNet B3 achieves slightly better performance when compared to Xception and Inception V3 models while using 10.6M and 11.6M fewer parameters, respectively. Also, Efficient B0 performs comparably to ResNet50 V2 with the same input image size while using approximately one-fifth of ResNet50 V2’s parameters.

Due to limited computing power, we could not train the B7 model of EfficientNet. If computational complexity is a concern, then models such as MobileNet and EfficientNet B0 can be trained to use considerably fewer parameters and still

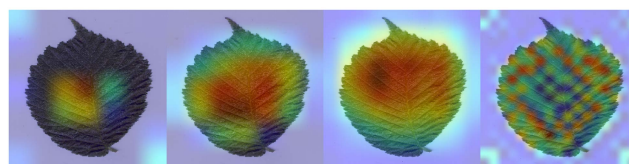


FIGURE 11. Grad-Cam visualizations of a leaf image for different networks. From left to right, MobileNetV2, EfficientNet B0, Xception, and EfficientNet B6, networks examine more detailed features as the number of parameters increase.

provide high accuracy and F-scores. Fig. 10 summarizes the number of parameters versus the F-score efficiency.

3) GRAD-CAM VISUALIZATION

In order to obtain gradient-weighted class activation mapping (Grad-CAM) visualization for any given class of image, the image is forward propagated through the CNN part of the model to obtain a raw score (without softmax activation) for the class. The gradient of the desired class is set to 1, and the remaining classes are set to 0. This signal is then backpropagated through the CNN to compute a heat map, which shows where the model had to look before making a decision. This heat-map then can be superimposed on the original image to create a Grad-CAM visualization [42]. Fig. 11 shows Grad-CAM visualizations for a leaf using the following four models from left to the right: MobileNet V2, EfficientNet B0, Xception, and EfficientNet B6. It can be observed from the Grad-CAM visualizations that as either the number of parameters or the input image size increases, models look at more details in an image before making a decision. The MobileNetV2 model primarily looks at the vein structure close to the midrib in the middle portion of the leaf, whereas EfficientNet B6 looks at significantly more detailed features such as detailed vein structure and the edge of the leaf before making a decision. This detailed feature extraction by the EfficientNet B6 model somewhat explains its superior performance compared to other models used in this paper.

4) COMPARISONS AND BENCHMARKS

The performance of the CNN architectures on the F2LSM dataset was bench-marked by comparing its performance

TABLE 6. State-of-the art accuracies on different datasets.

Dataset	Number of classes	Model	Top-1 accuracy (%)
Swedish [23]	15	Dual deep learning architecture [19]	99.41 [19]
Folio [24]	32	GoogleNet [11]	99.42 [46]
Flavia [22]	32	Extreme learning machines [47]	99.10 [47]
MEW 2012 [21]	153	SVM [48]	96.54 [49]
LeafSnap [20]	185	Siamese CNN [50]	96.00 [50]
F2LSM	374	EfficientNet B6 [15]	98.41

with the accuracy obtained on the individual datasets used to create the new dataset, as shown in Table 6. It can be observed from the table that the Top-1 accuracy generally drops as the number of classes increases. Even though F2LSM is a combined dataset with 374 classes (more than twice of LeafSnap), EfficientNet B6 achieves comparable accuracy compared to the models trained on the individual datasets.

D. DATA AND CODE AVAILABILITY

The resulting F2LSM dataset is available for download at https://scholarsmine.mst.edu/research_data/8/. The dataset includes individual links to each of the 374 folders for respective classes. The python code used to implement stratified k-fold cross-validation is also available on the website as a zipped file named *kFold.zip*. The python code takes '.csv' file with image addresses as an input for classification. The python code uses Pandas [43], Tensorflow [44], and Keras [45] libraries for implementation of stratified k-fold cross-validation of CNN architectures.

VI. CONCLUSION

This paper combines five publicly available leaf datasets into one F2LSM dataset. The combined dataset is highly imbalanced due to some of the individual datasets' imbalanced nature, some classes having very few image samples, and certain classes overlapping across different datasets while combining. We used oversampling and undersampling to mitigate the imbalance in the dataset. We then used TL to train several CNN architectures for plant species identification using leaf images in the F2LSM dataset and tested their performance using metrics such as precision, recall, and F-score, considering the imbalanced nature of the combined dataset. EfficientNet B6 achieved comparable accuracy on the F2LSM dataset compared to the state-of-the-art accuracy on the individual datasets. F2LSM dataset, and the python code to obtain the results presented in this paper, are available at the following website: https://scholarsmine.mst.edu/research_data/8/. Future work may include further expanding this dataset by including more plant species and using leaf images for other applications such as weed identification, plant phenotyping, and identification of leaf diseases and pests. Also, plant identification for occluded leaves in a field would be an interesting problem.

ACKNOWLEDGMENT

The authors would like to thank the reviewers and the editor for their constructive comments to help improve the quality of this paper.

REFERENCES

- [1] D. S. Fabricant and N. R. Farnsworth, "The value of plants used in traditional medicine for drug discovery," *Environ. Health Perspectives*, vol. 109, no. 1, pp. 69–75, 2001.
- [2] B. C. Bennett and M. J. Balick, "Does the name really matter? The importance of botanical nomenclature and plant taxonomy in biomedical research," *J. Ethnopharmacol.*, vol. 152, no. 3, pp. 387–392, Mar. 2014.
- [3] D. Rivera, R. Allkin, C. Obón, F. Alcaraz, R. Verpoorte, and M. Heinrich, "What is in a name? The need for accurate scientific nomenclature for plants," *J. Ethnopharmacol.*, vol. 152, no. 3, pp. 393–402, Mar. 2014.
- [4] R. W. Bussmann, "Taxonomy—An irreplaceable tool for validation of herbal medicine," in *Evidence-Based Validation of Herbal Medicine*. Amsterdam, The Netherlands: Elsevier, 2015, pp. 87–118.
- [5] B. C. Bennett and M. J. Balick, "Phytomedicine 101: Plant taxonomy for preclinical and clinical medicinal plant researchers," *J. Soc. Integrative Oncol.*, vol. 6, no. 4, p. 150, 2008.
- [6] Z. Wu, Y. Chen, B. Zhao, X. Kang, and Y. Ding, "Review of weed detection methods based on computer vision," *Sensors*, vol. 21, no. 11, p. 3647, May 2021.
- [7] Z. Li, R. Guo, M. Li, Y. Chen, and G. Li, "A review of computer vision technologies for plant phenotyping," *Comput. Electron. Agricult.*, vol. 176, Sep. 2020, Art. no. 105672.
- [8] J. Wäldchen and P. Mäder, "Plant species identification using computer vision techniques: A systematic literature review," *Arch. Comput. Methods Eng.*, vol. 25, no. 2, pp. 507–543, 2018.
- [9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1097–1105.
- [10] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis. Zürich, Switzerland: Springer*, 2014, pp. 818–833.
- [11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
- [12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
- [13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
- [14] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 1, pp. 4780–4789.
- [15] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.
- [16] J. Wäldchen, M. Rzanny, M. Seeland, and P. Mäder, "Automated plant species identification—Trends and future directions," *PLoS Comput. Biol.*, vol. 14, no. 4, 2018, Art. no. e1005993.
- [17] K. Pankaja and G. Thippeswamy, "Survey on leaf recognition and classification," in *Proc. Int. Conf. Innov. Mech. Ind. Appl. (ICIMIA)*, Feb. 2017, pp. 442–450.
- [18] O. H. Babatunde, L. Armstrong, J. Leng, and D. Diepeveen, "A survey of computer-based vision systems for automatic identification of plant species," *J. Agricult. Informat.*, vol. 6, no. 1, pp. 61–71, Jan. 2015.
- [19] A. P. Sundara Sobitha Raj and S. K. Vajravelu, "DDL: Dual deep learning architecture for classification of plant species," *IET Image Process.*, vol. 13, no. 12, pp. 2176–2182, Oct. 2019.
- [20] N. Kumar, P. N. Belhumeur, A. Biswas, D. W. Jacobs, W. J. Kress, I. C. Lopez, and J. V. B. Soares, "Leafsnap: A computer vision system for automatic plant species identification," in *Proc. Eur. Conf. Comput. Vis. Florence, Italy: Springer*, 2012, pp. 502–516.
- [21] P. Novotný and T. Suk, "Leaf recognition of woody species in central Europe," *Biosyst. Eng.*, vol. 115, no. 4, pp. 444–452, 2013.
- [22] 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 *Proc. IEEE Int. Symp. Signal Process. Inf. Technol.*, Dec. 2007, pp. 11–16.
- [23] O. Söderkvist, "Computer vision classification of leaves from Swedish trees," M.S. thesis, Linköping Univ., Linköping, Sweden, Sep. 2001.
- [24] T. Munisami, M. Ramsum, S. Kishnah, and S. Pudaruth, "Plant leaf recognition using shape features and colour histogram with K-nearest neighbour classifiers," *Proc. Comput. Sci.*, vol. 58, pp. 740–747, Jan. 2015.
- [25] C. X. Ling and C. Li, "Data mining for direct marketing: Problems and solutions," in *Proc. KDD*, vol. 98, 1998, pp. 73–79.

- [26] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explor. Newslett.*, vol. 6, no. 1, p. 20–29, Jun. 2004.
- [27] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla, "SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary," *J. Artif. Intell. Res.*, vol. 61, pp. 863–905, Apr. 2018.
- [28] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Proc. Int. Conf. Artif. Neural Netw.* Rhodes, Greece: Springer, 2018, pp. 270–279.
- [29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
- [30] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.
- [31] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
- [32] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.
- [33] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.
- [34] A. Chaudhury and J. L. Barron, "Plant species identification from occluded leaf images," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 17, no. 3, pp. 1042–1055, May 2020.
- [35] M. Kumar, S. Gupta, X.-Z. Gao, and A. Singh, "Plant species recognition using morphological features and adaptive boosting methodology," *IEEE Access*, vol. 7, pp. 163912–163918, 2019.
- [36] J. Huixian, "The analysis of plants image recognition based on deep learning and artificial neural network," *IEEE Access*, vol. 8, pp. 68828–68841, 2020.
- [37] X. Wang, W. Du, F. Guo, and S. Hu, "Leaf recognition based on elliptical half Gabor and maximum gap local line direction pattern," *IEEE Access*, vol. 8, pp. 39175–39183, 2020.
- [38] Y. Song, F. He, and X. Zhang, "To identify tree species with highly similar leaves based on a novel attention mechanism for CNN," *IEEE Access*, vol. 7, pp. 163277–163286, 2019.
- [39] V. Gajjar, Z. H. Lai, and K. Kosbar, "Fast classification of leaf images for agricultural remote sensing applications," in *Proc. Int. Telemetering Conf.*, Glendale, AZ, USA, vol. 54, Nov. 2018, pp. 569–578.
- [40] J. W. Tan, S.-W. Chang, S. Abdul-Kareem, H. J. Yap, and K.-T. Yong, "Deep learning for plant species classification using leaf vein morphometric," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 17, no. 1, pp. 82–90, Jun. 2018.
- [41] X. Zeng and T. R. Martinez, "Distribution-balanced stratified cross-validation for accuracy estimation," *J. Exp. Theor. Artif. Intell.*, vol. 12, no. 1, pp. 1–12, 2000.
- [42] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.
- [43] W. McKinney *et al.*, "Data structures for statistical computing in Python," in *Proc. 9th Python Sci. Conf.*, vol. 445, no. 1. Austin, TX, USA: SciPy, 2010, pp. 51–56.
- [44] M. Abadi. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: <https://www.tensorflow.org/>
- [45] F. Chollet (2015). *Keras*. [Online]. Available: <https://github.com/fchollet/keras>
- [46] P. Pawara, E. Okafor, L. Schomaker, and M. Wiering, "Data augmentation for plant classification," in *Proc. Int. Conf. Adv. Concepts Intell. Vis. Syst.* Antwerp, Belgium: Springer, 2017, pp. 615–626.
- [47] M. Turkoglu and D. Hanbay, "Recognition of plant leaves: An approach with hybrid features produced by dividing leaf images into two and four parts," *Appl. Math. Comput.*, vol. 352, pp. 1–14, Jul. 2019.
- [48] E. Osuna, R. Freund, and F. Girosi, "An improved training algorithm for support vector machines," in *Proc. Neural Netw. Signal Process. VII IEEE Signal Process. Soc. Workshop*, Sep. 1997, pp. 276–285.
- [49] C. Reul, M. Toepfer, and F. Puppe, "Cross dataset evaluation of feature extraction techniques for leaf classification," *International J. Artif. Intell. Appl. (IJAAI)*, vol. 7, no. 2, pp. 1–20, 2016.
- [50] V. M. Araújo, A. S. Britto Jr., L. S. Oliveira, and A. L. Koerich, "Two-view fine-grained classification of plant species," *Neurocomputing*, vol. 467, pp. 427–441, Jan. 2022.



VIRAJ K. GAJJAR (Graduate Student Member, IEEE) received the B.E. degree in electronics and communication from the Indus Institute of Technology and Engineering (Gujarat Technological University), Ahmedabad, Gujarat, India, in 2014. He is currently pursuing the Ph.D. degree in electrical engineering with the Missouri University of Science and Technology, Rolla, MO, USA. His research interests include computer vision, wireless communications, statistical signal analysis, machine learning, and deep learning.



ANAND K. NAMBISAN (Graduate Student Member, IEEE) received the B.Tech. degree in electronics and communication from Amrita Vishwa Vidyapeetham, Kerala, India, in 2013. He is currently pursuing the Ph.D. degree in electrical engineering with the Missouri University of Science and Technology, Rolla, MO, USA. His research interests include stochastic signal analysis, computer vision, and machine learning, especially dealing with problems of data imbalance and generalization.



KURT L. KOSBAR (Member, IEEE) received the B.S. degree in electrical and computer engineering from Oakland University, in 1982, and the M.S. and Ph.D. degrees in electrical engineering from the University of Southern California, in 1984 and 1988, respectively. He was a Staff Engineer with the Space and Communications Group, Hughes Aircraft Company, from 1982 to 1988. He is currently an Associate Department Chair with the Electrical and Computer Department, Missouri University of Science and Technology, Rolla, MO, USA. His research interests include signal and image processing, communication and telemetry systems, along with synchronization and navigation technology.

...