Summer 2022

# Machine learning applications in plant identification, wireless channel estimation, and gain estimation for multi-user software-defined radio

Viraj K. Gajjar

# MACHINE LEARNING APPLICATIONS IN PLANT IDENTIFICATION, WIRELESS CHANNEL ESTIMATION, AND GAIN ESTIMATION FOR MULTI-USER SOFTWARE-DEFINED RADIO

by

VIRAJ KISHORKUMAR GAJJAR

A DISSERTATION

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

2022

Approved by:

Dr. Kurt Kosbar, Advisor
Dr. Randy Moss
Dr. Akim Adekpedjou
Dr. Jiangfan Zhang
Dr. Sahra Sedigh Sarvestani

# PUBLICATION DISSERTATION OPTION

This dissertation consists of the following five articles, formatted in the style used by the Missouri University of Science and Technology.

Paper I: Fast Classification of Leaf Images for Agricultural and Remote Sensing Applications, found on Pages 8-21, has been accepted by the International Telemetry Conference (ITC).

Paper II: Plant Identification in a Combined-Imbalanced Leaf Dataset, found on Pages 22-50, has been accepted in the IEEE Access Journal.

Paper III: CSI Estimation Using Artificial Neural Network, found on Pages 51-62, has been accepted by the International Telemetry Conference (ITC).

Paper IV: Rapid Gain Estimation for Multi-User Software Defined Radio Applications, found on Pages 63-73, has been accepted by the International Telemetry Conference (ITC).

Paper V: Distribution-Based Gain Estimation for Aeronautical Software-Defined Radio Applications, found on Pages 74-93, is submitted to IEEE Transactions on Aerospace and Electronic Systems.

# ABSTRACT

This work applies machine learning (ML) techniques to selected computer vision and digital communication problems. Machine learning algorithms can be trained to perform a specific task without explicit programming. This research applies ML to the problems of: plant identification from images of leaves, channel state information (CSI) estimation for wireless multiple-input-multiple-output (MIMO) systems, and gain estimation for a multi-user software-defined radio (SDR) application.

In the first task, two methods for plant species identification from leaf images are developed. One of the methods uses hand-crafted features extracted from leaf images to train a support vector machine classifier. The other method combines five publicly available leaf datasets: Flavia, Folio, LeafSnap, Swedish, and Middle European Woods 2014, to create a new data set named F2LSM. To create a benchmark, multiple end-to-end convolutional neural network classifiers are trained to classify images in the F2LSM dataset.

The second application of ML is a novel CSI estimation technique for MIMO communication systems. The approach uses atmospheric conditions, the position of the transmitter and receiver, and the relative motion of the transmitter and receiver as features for an artificial neural network (ANN).

The third study uses two ML methods to estimate gain for a multi-user SDR system in an aircraft, where a single SDR must generate a composite waveform for multiple communication links. An accurate estimate of the composite waveform's peak is required to set the digital-to-analog converter's gain to a value that will avoid clipping, while minimizing quantization noise. One of the methods uses an ANN to estimate the waveform's peak and statistical moments. The other method uses an ANN to estimate the statistical distribution parameters that closely represent the voltage distribution of the waveform.

## ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Kurt Kosbar, for his support throughout the years. Dr. Kosbar found unique and challenging opportunities and gave me absolute freedom to pursue these projects while providing excellent insights. I aspire to be half as enthusiastic, knowledgeable, and lively as Dr. Kosbar and hopefully someday be able to command an audience as well as he does. I would like to express my gratitude to my doctoral research committee members, Dr. Randy Moss, Dr. Akim Adekpedjou, Dr. Jiangfan Zhang, and Dr. Sahra Sedigh Sarvestani, for their time and valuable guidance. My special thanks to my friends and colleagues, Anand Nambisan and Ze-Hao Lai, for helping me in my research. I would finally like to thank my family– my parents, Kishor Gajjar and Rita Gajjar; my brother, Akshay; and my friends for their continued love and support. I would not have made it this far without them.

## TABLE OF CONTENTS

SECTION

## LIST OF ILLUSTRATIONS

Figure Page

PAPER I

PAPER II

PAPER III

PAPER IV

PAPER V

## LIST OF TABLES

Table Page

PAPER II

PAPER III

PAPER IV

PAPER V

# SECTION

# 1. INTRODUCTION

The primary domains of machine learning (ML) applications involve: computer vision, prediction, semantic analysis, natural language processing, and information retrieval [101]. The motivation of this work is to apply ML techniques in computer vision for plant identification, and in prediction for wireless channel estimation for a multiple-input-multiple-output (MIMO) system and gain estimation for multi-user software-defined radio (SDR).

## 1.1. INTRODUCTION TO MACHINE LEARNING

ML is the ability to train a computer to perform a task without explicitly programming it. ML has observed rapid growth since Geoffrey Hinton et al. [42] introduced deep learning (DL) to recognize handwritten digits. DL is an ML algorithm that extracts features from raw data using multiple layers; deeper layers progressively extract a higher level of features. DL models are based on artificial neural networks (ANN) with feature learning, and are inspired by biological neural networks.

ML systems can be classified into the following four major categories based on the amount and type of supervision they get during training [39]: supervised learning (SL), unsupervised learning, semi-supervised learning, and reinforcement learning. In this dissertation, we primarily focus on supervised learning, where the training data is in the form of input-output pairs, i.e., for each input vector, there is the desired output vector. An SL algorithm will find a function that maps input vectors to output vectors based on the training data.

Supervised ML algorithms are often used for classification and regression. Consider a dataset $D$ of $N$ input-output pairs, $D = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), ..., (\mathbf{x}^{(N)}, \mathbf{y}^{(N)})\}$. The inputs of $D$ are $p$ descriptive features given as, $\mathbf{x}^{(r)} = (x_1^{(r)}, ..., x_i^{(r)}, ..., x_p^{(r)})$, where $i \in \{1, ..., p\}$ and $r \in \{1, ..., N\}$. For a classification task $\mathbf{y}^{(r)}$ is the $r^{th}$ label and for a regression task $\mathbf{y}^{(r)}$ is the $r^{th}$ continuous target. In case of multi-output regression, $\mathbf{y}^{(r)} = (y_1^{(r)}, ..., y_j^{(r)}, ..., y_q^{(r)})$ is a vector with $q$ continuous targets, $j \in \{1, ..., q\}$.

Classification models like convolutional neural networks (CNN) and support vector machines (SVM) are used to identify plant species based on leaf images. Multi-output regression algorithms like ANNs are used to address channel and gain estimation problems in digital communications.

### 1.2. MACHINE LEARNING IN PLANT IDENTIFICATION

ML has been widely used for plant phenotyping [65, 75, 123], plant identification [5, 82, 124, 125], and in precision agriculture for plant disease detection and weed identification [66, 70, 96, 100, 102, 137].

Plant identification is essential in plant taxonomy and botanical nomenclature, as it can help avoid errors and ambiguities in phytomedicine, ethnopharmacology, and other research on plants [10, 11, 14, 92]. Therfore, phytological research can benefit from an automated plant identification system that is automated and reliable. Leaves are one of the most distinctive plant features and are widely used for plant identification. More than 100 studies have used images of leaves to identify plants [124]. This work primarily focuses on plant identification using leaf images obtained from publicly available leaf datasets.

ML algorithms traditionally require handcrafted (HC) features for image classification. HC features requires a human expert to define the features that could be potentially helpful for ML systems to identify an image's class. These features are then fed to a ML classifier to make a classification decision. Since DL algorithms such as CNNs do not require a user to define features, the models automatically identify and extract useful features for a specific classification task. DL models that identify and extract features generally outperform classifiers that use HC features. Since DL models require a large amount of training data, HC features can be useful when training data is insufficient [67]. HC feature extraction, end-to-end DL classifier models, and hybrid techniques combining DL and HC feature extraction have been widely used for plant-leaf identification.

Kumar et al. [59] proposed a leaf dataset of plant species found in the Northeastern United States and Canada, and a mobile application, LeafSnap, to classify images. The application used histogram of curvature over scale features to train a nearest-neighbor classifier. Zulkifli et al. [141] demonstrated the effectiveness of Zernike moment invariant, Legendre moment invariant, and Tchebichef moment invariant features in extracting features from leaf images using a general regression neural network. Novotný and Suk [80] used Fourier descriptors for leaf contours combined with the nearest-neighbor classifier to classify leaf images in the proposed Middle European Woods 2012 dataset. Wang et al. [127] extracted shape features of leaves and used a hypersphere classifier to classify leaf images. Chaudhury et al. [15] used the concept of contour-based 2D shape matching to identify plant species from occluded leaf images by using the contour information. Wu et al. [135] proposed the Flavia leaf dataset and used twelve leaf-shape features to train a probabilistic neural network for classification.

Multiple end-to-end DL models have also been proposed for plant identification. Lee et al. [62] use CNNs to perform unsupervised feature extraction for classification, and provide a technique to visualize features using deconvolutional networks. Song et al. [109] proposed a novel attention branch-based convolutional neural network (ABCNN) to identify plant species with highly similar leaves. Raj et al. [88] proposed a dual deep learning architecture that combines two CNN models, MobileNet [98] and DenseNet-121 [43], for feature extraction and classification. [5, 82, 119, 124, 125] are detailed surveys of plant identification techniques using leaf images.

## 1.3. MACHINE LEARNING FOR CHANNEL AND GAIN ESTIMATION

ML's remarkable success in recent years has led to an increased interest in its application in communication systems [21, 49]. Multiple ML techniques have been implemented for radar radiation source classification, and recognition, radar image processing, and anti-jamming and interference mitigation [61]. Peng et al. [84] use signal constellation diagrams, whereas Meng et al. [74] directly use received signals' complex baseband representation as features to train a CNN for automatic modulation classification. Wang et al. [126] use reinforcement learning to design an intelligent waveform optimization method for multitarget detection of MIMO radar. Ye et al. [138] demonstrate a way to directly recover transmitted symbols in an orthogonal frequency division multiplexing system using DL without having to estimate channel state information (CSI) explicitly. Donohoo et al. [27] demonstrated a way to learn a mobile terminal's usage pattern in a cellular system to provide optimal handover solutions. ML has been used to improve pilot contamination in cellular massive MIMO by learning channel parameters of the desired link in the target cell, and undesired link in the adjacent cell [129]. This dissertation focuses on training ML models, specifically ANNs, to estimate CSI for a MIMO system and digital-to-analog converter's gain for a multi-user SDR.

Least squared (LS) and minimum mean square error (MMSE) methods have been conventionally used to estimate CSI. LS estimation requires no information about the channel's statistics compared to MMSE, which performs better by utilizing channel statistics. DL has recently gained attention in channel estimation [40, 72, 108]. Soltani et al. [108] use pilot values as a low-resolution image to train a cascaded network of image super-resolution and image restoration for channel estimation. In [72], a combination of CNN and long short-term memory network is used to estimate CSI in a 5G wireless communication system using features that affect CSI in combination with CSI data sample as a feature. He et al. [40] demonstrates a channel estimation in beamspace millimeter-wave MIMO systems using a neural network that learns channel structure when the receiver is equipped with a limited number of radio-frequency chains. Bai et al. [6] propose a DL-based channel estimator under a time-varying Rayleigh fading channel that can dynamically track the channel status without prior knowledge about the channel model and statistics characteristics. [18] proposes a DL-based CSI estimator for a MIMO system for a particular case when the length of the pilot signal is smaller than the number of transmit antennas.

Commercial aircrafts require multiple connectivity applications such as air traffic management, aircraft communication addressing and reporting, single-pilot Operations, and massive data transfer [7]. These applications are facilitated through direct air-to-ground communications, satellite networks, high-altitude platform networks, aircraft-to-aircraft links, and multi-link networks [7]. The communication links associated with these applications use cabling and hardware to generate baseband waveforms. It can be challenging to integrate new applications on a hardware system, and to modify, integrate, troubleshoot, and maintain them due to cabling [97]. Using software to generate the baseband waveforms instead of hardware can help address these concerns as it can reduce the amount of hardware and cabling. An SDR can be an excellent fit for this since it can support multiple radio bands with software versatility. An SDR can essentially either generate or take digital samples and represent them as baseband radio waves, which can then be up-converted and transmitted using transceiver hardware [86]. Amrhar et. al. demonstrated that an SDR-based integrated modular avionics architecture could significantly reduce the size, weight, power, and cost in an aircraft [2]. However, if a single SDR generates waveforms for multiple applications, estimating the gain of a digital-to-analog converter (DAC) for the combined waveform becomes challenging. A multi-output regression algorithm can potentially help estimate DAC's gain based on the features of the component signals.

## 1.4. ORGANIZATION OF THE DISSERTATION

In this dissertation, three applications of ML are explored: plant identification, CSI estimation, and gain estimation for multi-user SDR. Five papers are presented for these applications.

The first paper presents a plant identification method using an SVM classifier. The features presented to the SVM classifier are HC features extracted from leaf images. A combination of binary image features, gray-scale image features, the gradient of an image features, RGB color-space features, YCbCr color-space features, and HSV color-space features are used to achieve state-of-the-art accuracy using HC features on two publicly available leaf datasets, Flavia and Swedish.

The second paper proposes a combined leaf dataset, F2LSM, created by combining five publicly available leaf datasets, Flavia, Folio, LeafSnap, Swedish, and Middle European Woods (MEW) 2014. The F2LSM dataset is available for public download at `https://scholarsmine.mst.edu/research_data/8/`. The combined dataset is imbalanced, so oversampling and undersampling are implemented to mitigate the effect of imbalance while training an ML model. Multiple CNN models are trained for leaf identification in the F2LSM dataset using transfer learning to set up benchmarks for future comparisons. The F2LSM dataset is made publicly available for further research.

The third paper proposes a CSI estimation technique for a MIMO system using an ANN. The model of the MIMO system is limited to line-of-sight communication without multipath. A dataset of CSI matrices is generated to train the ANN for varying atmospheric conditions, and varying velocities and positions of transmitters and receivers. Once trained, the ANN can estimate CSI using the following features: rain rate, atmospheric fog density, humidity, temperature, the distance between transmitter and receiver, receiver's velocity relative to the transmitter, carrier frequency, and angle elevation from transmitter to receiver.

The fourth paper proposes a rapid gain estimation technique for aeronautical multi-user SDR applications. An SDR can generate multiple baseband waveforms for different communication applications in an aircraft. It can be challenging to determine the peak value of the combined waveform generated by an SDR. An ANN-based technique is presented to estimate the peak and statistics of the combined waveform. The statistics estimated are mean, variance, skewness, kurtosis, and fifth and standardized moments. The statistics and the estimated peak should allow a user to choose the optimal gain for a DAC.

The fifth paper is an extension of the fourth paper. In the fifth paper, the voltage distribution of the combined signal is estimated. It is shown that the combined signal follows a generalized gamma distribution (GGD). In this paper, ANN is used to estimate the shape and scale parameters of the GGD.

# PAPER

## I. FAST CLASSIFICATION OF LEAF IMAGES FOR AGRICULTURAL REMOTE SENSING APPLICATIONS

Viraj K. Gajjar. Ze -H. Lai, and Kurt L. Kosbar
Department of Electrical Engineering
Missouri University of Science and Technology
Rolla, Missouri 65409–0050
Tel: 573–202–1062
Email: vgf4c@mst.edu

## ABSTRACT

This paper introduces a method of classifying leaves using machine learning. Considerable emphasis has been put on leaf classification for use in remote sensing applications such as plant phenotyping and precision agriculture. Convolutional neural networks (CNN) have been extensively used in computer vision for image classification. However, CNN can be computationally expensive. This paper describes a method that achieves a comparable accuracy, with a lower computational burden, using a support vector machine (SVM) classifier. This method uses image processing algorithms to extract features from Hough transform and Hough Lines. These features are then integrated with those extracted from binary images, and "eigenleaves" extracted from grayscale, gradient, and different color-space images of leaves as data preprocessing for classification. The classifier is implemented on two publicly available datasets: Flavia and Swedish; and is able to achieve state-of-the-art accuracies using a SVM classifier.

**Keywords:** leaf classification, support vector machines, feature extraction, eigenleaves

# 1. INTRODUCTION

Precise identification of leaves has applications such as site-specific weed management in agriculture and plant phenotyping. With a growing population, there is an increasing demand for efficient agricultural methods. The ability to identify weeds in the field using an autonomous robot, perhaps one which has a robotic arm to remove weeds, or a drone which can spray herbicide on the desired area could be helpful. Such automated systems need a way to differentiate desirable plants from undesirable weeds. One way to do this is with an imaging system that can recognize different species of plants. Some recent machine algorithms use convolutional neural networks (CNN) for this operation. However, they are computationally expensive operations. This paper proposes using a SVM classifier instead of using CNN, which can substantially reduce the computational burden. This new classifier was trained using two publicly available datasets named Flavia and Swedish. The classifier was able to analyze images using the processing power available on conventional agricultural drones. There was no need to augument the processor with a graphics processing unit (GPU) or similar hardware, nor was there a need to relay data to a ground station for processing.

A way to improve the accuracy of leaf classification by training auxiliary data for classification of leaves when training dataset is small has been previously shown by Wu [1]. This method compares leaf shapes by aligning their angle sequences to one another, rather than extracting features from leaf shapes. It was shown by Ling [2] that using inner-distance as a descriptor for an image can be helpful in improving the performance of identification of leaves. Im [3] and Wang [4] used hierarchical polygon approximation representation for leaf shape and hypersphere classifier respectively to classify leaves. Segmentation of leaves from the background using 3-D point cloud was done by Teng [5], which would be a bridge between the system for capturing the images and classifying them. Elliptic Fourier transforms were used to identify weed species by Neto [6] with an accuracy of 89.4%.

Our proposed method extracts distinctive features from images in binary, monochrome (grayscale and gradient) and different color-spaces (RGB, HSV, and YCbCr). Binary images are used to extract binary object features such as area and perimeter of the object. Grayscale images are used to calculate Gray-Level Co-occurrence Matrix (GLCM) from which feature vectors are extracted. Grayscale images are also used to calculate Hough transform and Hough lines which help in extracting features related to the vein structures and the contour of the leaf. We find the number of corners in a leaf image using the method described by Rosten [7], the number of speeded up robust features (SURF) points using the method described in Bay [8], and the number of binary robust invariant scalable keypoints (BRISK) using the method described by Leutenegger [9]. Using the entire feature vectors instead of using the number of points can be helpful in improving the accuracy of the system, but as the feature vectors are very large it would be computationally expensive. Later we show that we can achieve the desired accuracy even without using the entire feature vectors of BRISK and SURF. The grayscale images along with the all the different color-space images are used to extract eigenleaves which are eigenvectors of the leaves projected on the leaf images. The idea of eigenleaves is inspired from [10] where the authors use eigenfaces to classify faces, where eigenfaces are the eigenvectors of the set of faces.

All these features extracted from the images are then used to train a SVM classifier. The features are also tested on various other classifiers too to test the accuracy. By using the hand-crafted features, and SVM classifier, we can achieve accuracy rates of 97.71% on the Flavia dataset [11] and accuracy of 99.56% on the Swedish dataset [12].

## 2. PREVIOUS WORK

Wu [11] developed a leaf recognition algorithm using a probabilistic neural network (PNN) and achieved an accuracy of 90% on the Flavia dataset. Singh [13] was able to achieve 96% accuracy on the Flavia dataset using SVM based on a binary decision tree. Kadir [14] used principal component analysis (PCA) to reduce the dimension of the features and improve the performance of classification using PNN. Nguyen [15] combines SURF features with bag-of-words to achieve an accuracy of 95.94% on the Flavia dataset. Kumar [16] developed the first android classification application LeafSnap which used leaf's contour over multiple scales for classification. Kadir [17] uses Zernike moments extracted from leaf images to train the PNN along with features extracted from gray level co-occurrence matrix and achieves an accuracy of 94.69% on the Flavia dataset. Wu [18] used CNN for classification of leaves using PReLU activation function and were able to get an accuracy of 94.8% on the ICL dataset with 25 different species of leaves. Wick showed in [19] that almost perfect accuracy can be achieved on Flavia dataset using Deep CNN.

## 3. PROPOSED METHOD

Our method can be divided into two major steps: Extraction of features from images, and training and testing of the classifier. The extraction of features is summarized in Figure 1.

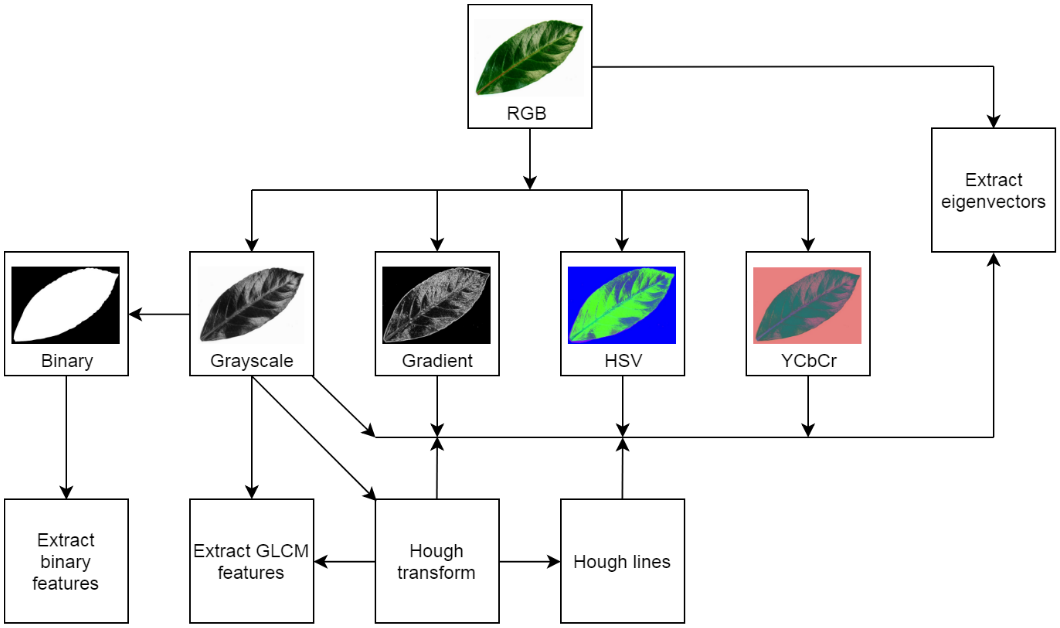


Figure 1. Features extraction.

## 3.1. EXTRACTING FEATURES FROM THE BINARY IMAGE

To extract features from a binary image, the RGB image of the leaf is first converted into a grayscale image and then to binary. Then some morphological operations such as closing, removing objects smaller than a certain area, and filling the holes are applied to the binary image to get rid of small erroneous objects and to get leaf as the only object in the image. Figure 2 summarizes the preprocessing step.

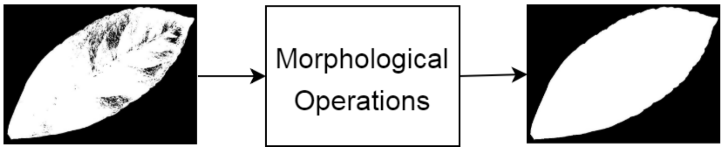


Figure 2. Performing morphological operations on the binary image to remove errors.

Once the preprocessing step is done following features are extracted from the binary images:

1. Eccentricity

   Since the shape of a leaf resembles an ellipse, eccentricity can be used as a feature for classification. The eccentricity of an ellipse can be calculated as:

$$eccentricity = \frac{distance\ between\ two\ foci}{length\ of\ major\ axis} \tag{1}$$

2. Ratio of minor axis to major axis of an ellipse

3. Perimeter to area ratio

   This is the ratio of the distance around the boundary of the leaf object to the number of pixels in the region of the leaf object.

4. Solidity

   Solidity is defined as the proportion of the pixels in the convex hull which are also in the region. The solidity of an object can be calculated as:

$$solidity = \frac{Area}{Convex\ Area} \tag{2}$$

   where convex area can be defined as the number of pixels in the smallest convex polygon that can contain the object.

5. Centroid

   In this case, a centroid is the center of mass of the leaf object

6. Aspect Ratio

   Aspect ratio is defined as the ratio of the height of bounding box containing the object to the width of the bounding box. Aspect ratio can be calculated as:

$$Aspect\ ratio = \frac{height\ of\ the\ bounding\ box}{width\ of\ the\ bounding\ box} \tag{3}$$

## 3.2. EXTRACTING FEATURES FROM A GRAYSCALE IMAGE

The following features are extracted from grayscale images:

**3.2.1. GLCM Features.** Several features can be calculated from GLCM and four of them are used as features for training the classifier.

1. Contrast

   Calculates the contrast between each pixel and its neighbor and is given as,

$$Contrast = \sum_i \sum_j (i - j)^2 c(i, j) \tag{4}$$

   where $c(i, j)$ denotes an element in GLCM.

2. Correlation

   Calculates how correlated each pixel is to its neighboring pixel and is given as,

$$Correlation = \frac{(i - \mu i)(j - \mu j)c(i, j)}{\sigma_i \sigma_j} \tag{5}$$

   where $\mu$ and $\sigma$ denote mean and standard deviation respectively.

3. Energy

   Calculates the energy of each element in GLCM and is given as ,

$$Energy = c(i, j)^2 \tag{6}$$

4. Homogeneity

   Calculates the closeness of the distribution of elements in GLCM to the diagonal of GLCM and is given as,

$$Homogeneity = \frac{c(i, j)}{1 + |i - j|} \tag{7}$$

**3.2.2. Number of Corner Points.** Leaves from different species could have a different number of corner points so using the number of corners as a feature would help to boost up the accuracy of the system. We use the high-speed machine learning algorithm presented by Rosten [7] to calculate the number of corner points in a leaf image.

**3.2.3. Number of SURF and Brisk Points.** SURF features were introduced by Bay [8] which are scale- and rotation-invariant detector and descriptor. BRISK features were introduced by Leutenegger [9] which detects, describes and matches key points in a grayscale image. In this step, rather than using the entire description of the features vector, we just use the number of SURF and BRISK points in an image as a descriptor as the leaves from the same species should have a similar number of SURF and BRISK points.

**3.2.4. Hough Transform.** One of the most distinctive features of the leaves is the vein pattern. Hough transform is a technique that finds aligned points in an image that create, so it helps in detecting vein features of a leaf image. Hough transform can be applied to a grayscale image by detecting the edges in the image and then applying the Hough transform algorithm, as summarized in Figure 3.

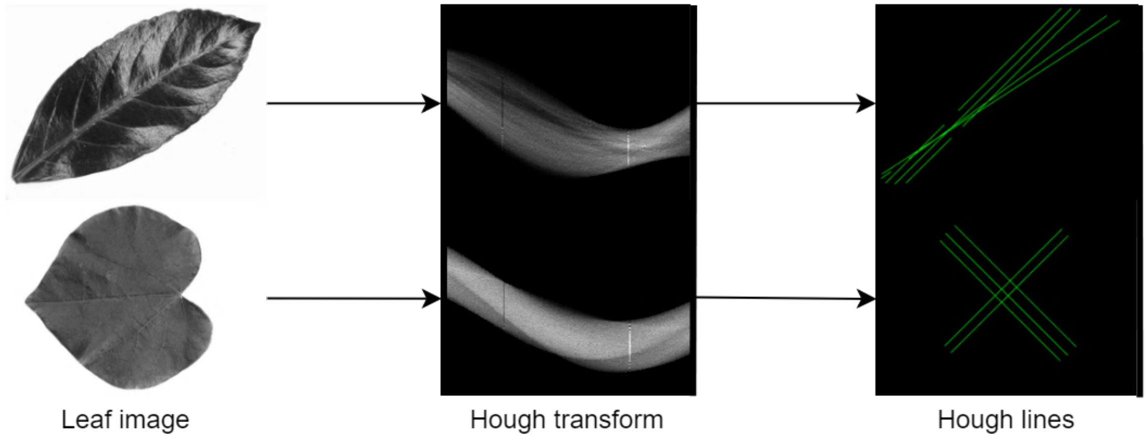


Figure 3. Hough transform and Hough lines of grayscale leaf images.

As it can be seen from Figure 3 that Hough transform of one leaf differs from another, so GLCM features of Hough transform images too are calculated.

**3.2.5. Hough Lines.** The most significant Hough peaks in the figure can be used to find Hough lines and as shown in Figure 3 different leaves have significantly different Hough lines.

### 3.3. EXTRACTING EIGENLEAVES

As seen from Figure 1, eigenleaves are extracted from seven different images. Eigenleaves are calculated for different color-spaces and for grayscale and gradient images following the steps described below:

1. Convert image into the desired colorspace

2. Vectorize all images and put in a matrix → $\mathbf{X}$

3. Calculate mean of corresponding pixels of each image → $\boldsymbol{\mu}$

4. Subtract the mean from all images → $\boldsymbol{\bar{X} = X - \mu}$

5. Perform eigen-decomposition of the of $\boldsymbol{X^T X}$ and get $K$ eigenvectors corresponding to $K$ largest eigenvalues → $\boldsymbol{v_k}$

6. Calculate eigenleaves → $\boldsymbol{l_k = \bar{X} v_k}$, where $k \in [1, K]$

Thus, there are eigenleaves of length $K$ associated with each leaf image which are used as features for training the classifier. The eigenleaves are also calculated for Hough transform and Hough lines images as they too provide distinctive features for training the classifier and helps boosting the accuracy.

## 4. EXPERIMENTS AND RESULTS

Our approach is tested on two of the publicly available datasets: Flavia and Swedish. Flavia dataset consists of 1907 leaf images of 32 different species and the Swedish dataset consists of 1125 images of 15 different species. SVM classifier is trained for both the datasets using the handcrafted features that are extracted from the leaf images. The classifier is tested on 10% images from the Flavia dataset and 20% images from the Swedish dataset respectively. Then we fine-tune training and testing data-size to get an optimal accuracy of 97.91% for the Flavia dataset and 99.56% for the Swedish dataset. Figure 4 shows how the accuracy of the system changes as we reduce the number of training images.

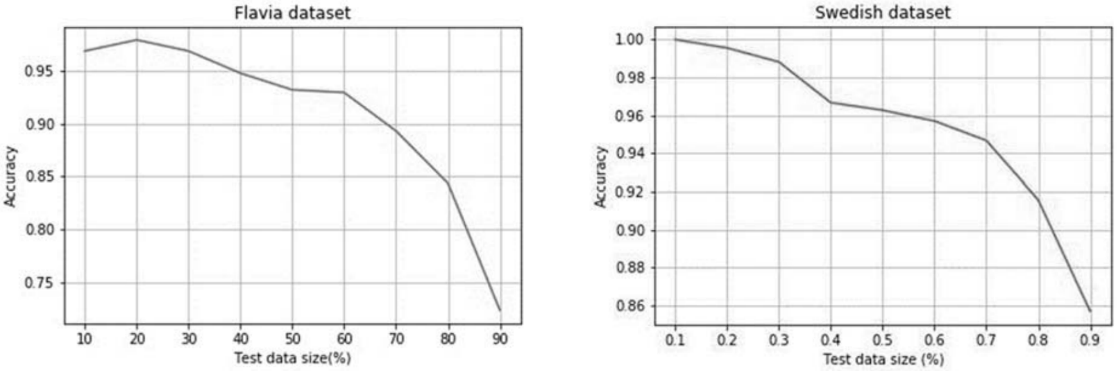
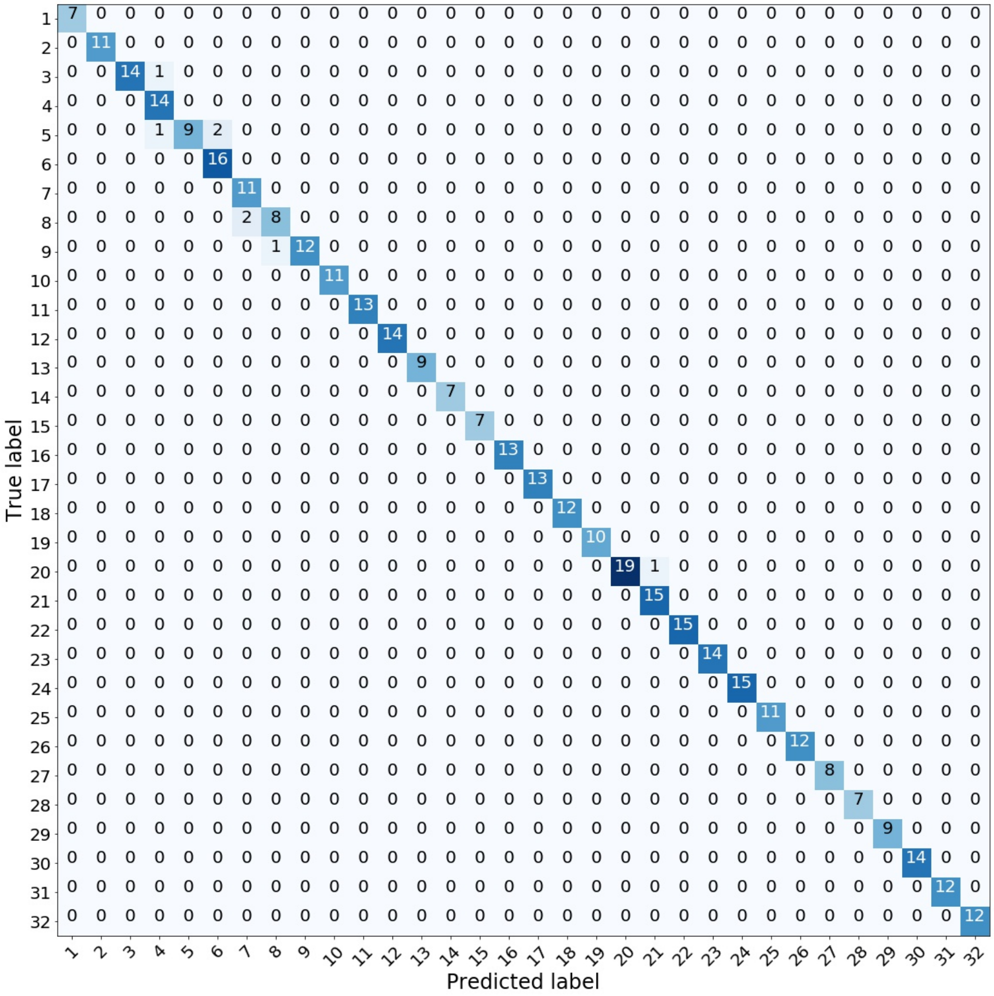
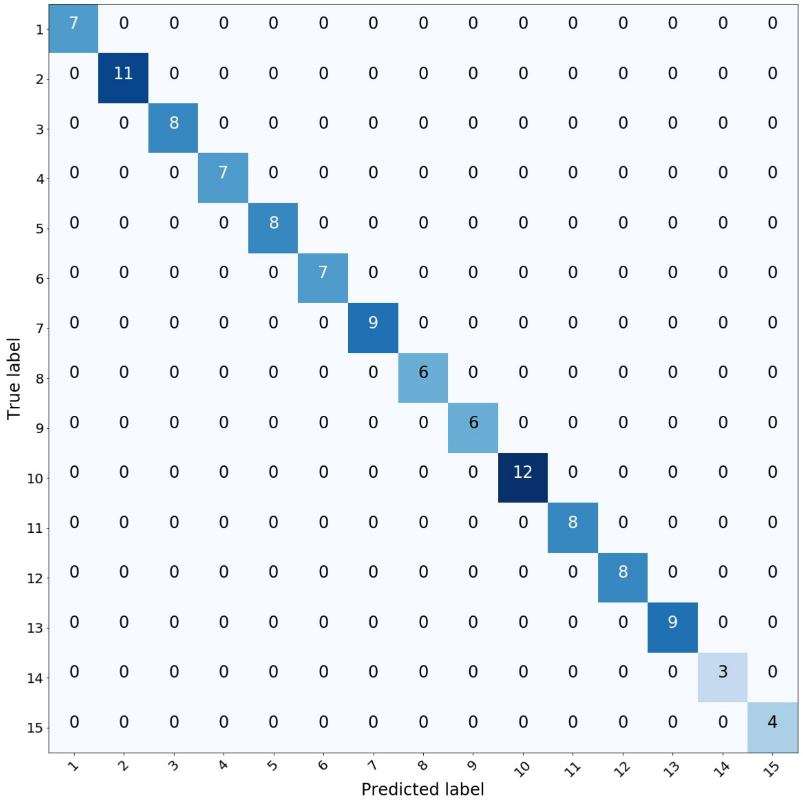


Figure 4. Accuracy of the SVM with changing test data size.



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 14 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 1 | 9 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 19 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 24 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 25 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 26 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 0 | 0 | 0 | 0 | 0 |
| 27 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 0 |
| 28 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 |
| 29 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 0 | 0 | 0 |
| 30 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14 | 0 | 0 |
| 31 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 12 | 0 |
| 32 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 12 |

Figure 5. Confusion matrix for Flavia dataset.



| True label \ Predicted label | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |

Figure 6. Confusion matrix for Swedish dataset.

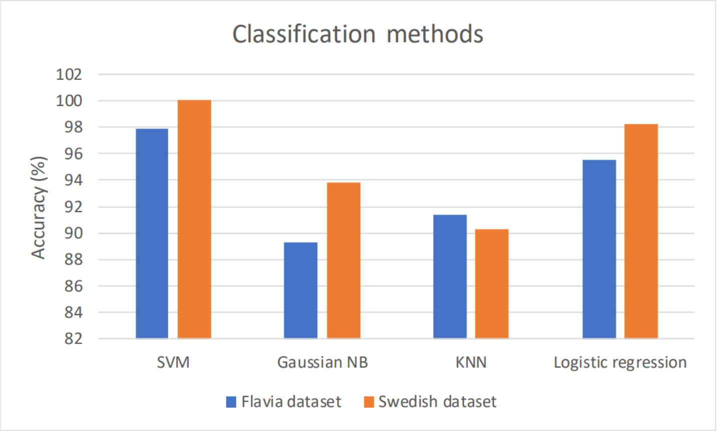


Figure 7. Comparison between different classification methods.

We also tested our features on several other classifiers, with the results summarized in Figure 7.

## 5. CONCLUSIONS

We are able to achieve state-of-the-art accuracy for Flavia and Swedish datasets using SVM classifier. Features are extracted from leaf object in binary images and features from GLCM are extracted by calculating GLCM for grayscale images. Grayscale images were also used to calculate Hough transform to extract GLCM features and Hough lines. Finally, eigenleaves are calculated from images in different color-spaces and from the monochrome images to be used as features. We are successfully able to get comparable accuracy to the state-of-the-art using a computationally efficient method. Furthermore, we calculate features from the Fast Fourier Transform (FFT) and Curvelet transform but those features are redundant with respect to the features we already have and are not able to improve the accuracy. The algorithm remains to be tested on other publicly available datasets namely: Foilage, LeafSnap, ICL, and ImageCLEF.

## REFERENCES

[1] Pengcheng Wu and Thomas G. Dietterich. Improving svm accuracy by training on auxiliary data sources. ICML '04, page 110, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015436. URL `https://doi.org/10.1145/1015330.1015436`.

[2] H. Ling and D.W. Jacobs. Using the inner-distance for classification of articulated shapes. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 719–726 vol. 2, 2005. doi: 10.1109/CVPR .2005.362.

[3] Cholhong Im, Hirobumi Nishida, and Tosiyasu L Kunii. A hierarchical method of recognizing plant species by leaf shapes. In *Proceedings of IAPR Workshop on Machine Vision Applications (MVA 1998)*, pages 158–161. Citeseer, 1998.

[4] Xiao-Feng Wang, Ji-Xiang Du, and Guo-Jun Zhang. Recognition of leaf images based on shape features using a hypersphere classifier. In De-Shuang Huang, Xiao-Ping Zhang, and Guang-Bin Huang, editors, *Advances in Intelligent Computing*, pages 87–96, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-31902-3.

[5] Chin-Hung Teng, Yi-Ting Kuo, and Yung-Sheng Chen. Leaf segmentation, its 3d position estimation and leaf classification from a few images with very close viewpoints. In Mohamed Kamel and Aurélio Campilho, editors, *Image Analysis and Recognition*, pages 937–946, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-02611-9.

[6] João Camargo Neto, George E. Meyer, David D. Jones, and Ashok K. Samal. Plant species identification using elliptic fourier leaf shape analysis. *Computers and Electronics in Agriculture*, 50(2):121–134, 2006. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2005.09.004. URL `https://www.sciencedirect.com/science/article/pii/S0168169905001560`.

[7] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 430–443, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-33833-8.

[8] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[9] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *2011 International Conference on Computer Vision*, pages 2548–2555, 2011. doi: 10.1109/ICCV.2011.6126542.

[10] Matthew A Turk and Alex P Pentland. Face recognition using eigenfaces. In *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 586–587. IEEE Computer Society, 1991.

[11] Stephen Gang Wu, Forrest Sheng Bao, Eric You Xu, Yu-Xuan Wang, Yi-Fan Chang, and Qiao-Liang Xiang. A leaf recognition algorithm for plant classification using probabilistic neural network. In *2007 IEEE International Symposium on Signal Processing and Information Technology*, pages 11–16, 2007. doi: 10.1109/ISSPIT.2007.4458016.

[12] Oskar Söderkvist. Computer vision classification of leaves from Swedish trees. In *MS Thesis*. Linköping University, Linköping, Sweden, 2001.

[13] Krishna Singh, Indra Gupta, and Sangeeta Gupta. Svm-bdt pnn and fourier moment technique for classification of leaf shape. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 3(4):67–78, 2010.

[14] Abdul Kadir, Lukito Edi Nugroho, Adhi Susanto, and Paulus Insap Santosa. Performance improvement of leaf identification system using principal component analysis. *International Journal of Advanced Science and Technology*, 44(11):113–124, 2012.

[15] Quang-Khue Nguyen, Thi-Lan Le, and Ngoc-Hai Pham. Leaf based plant identification system for android using surf features in combination with bag of words model and supervised learning. In *2013 International Conference on Advanced Technologies for Communications (ATC 2013)*, pages 404–407, 2013. doi: 10.1109/ATC.2013.6698145.

[16] Neeraj Kumar, Peter N. Belhumeur, Arijit Biswas, David W. Jacobs, W. John Kress, Ida C. Lopez, and João V. B. Soares. Leafsnap: A computer vision system for automatic plant species identification. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 502–516, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-33709-3.

[17] Abdul Kadir, Lukito Nugroho, Adhi Susanto, and Paulus Santosa. Experiments of zernike moments for leaf identification. *Journal of Theoretical and Applied Information Technology*, 41:82–93, 07 2012.

[18] Yan-Hao Wu, Li Shang, Zhi-Kai Huang, Gang Wang, and Xiao-Ping Zhang. Convolutional neural network application on leaf classification. In De-Shuang Huang, Vitoantonio Bevilacqua, and Prashan Premaratne, editors, *Intelligent Computing Theories and Application*, pages 12–17, Cham, 2016. Springer International Publishing. ISBN 978-3-319-42291-6.

[19] Christoph Wick and Frank Puppe. Leaf identification using a deep convolutional neural network, 2017. URL `https://arxiv.org/abs/1712.00967`.

# II. PLANT IDENTIFICATION IN A COMBINED-IMBALANCED LEAF DATASET

Viraj K. Gajjar. Anand K. Nambisan, and Kurt L. Kosbar
Department of Electrical Engineering
Missouri University of Science and Technology
Rolla, Missouri 65409–0050
Tel: 573–202–1062
Email: 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.

**Keywords:** leaf dataset, imbalanced dataset, convolutional neural networks, transfer learning, plant identification

# 1. 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, 3, 4, 5].

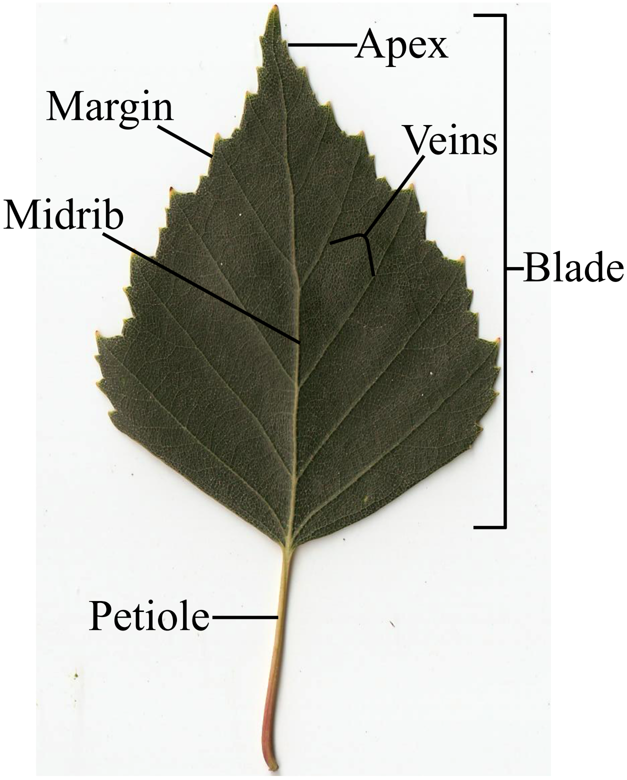


Figure 1. Parts of leaves that can be potentially used as features by computer vision algorithms for classification.

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, 3, 4, 5]. A faster, automated, and more reliable method of plant species identification would be helpful in phytological 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 Figure 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, 10, 11, 12, 13, 14, 15]. A CNN is able to extract features in 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, 17, 18, 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, 26, 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

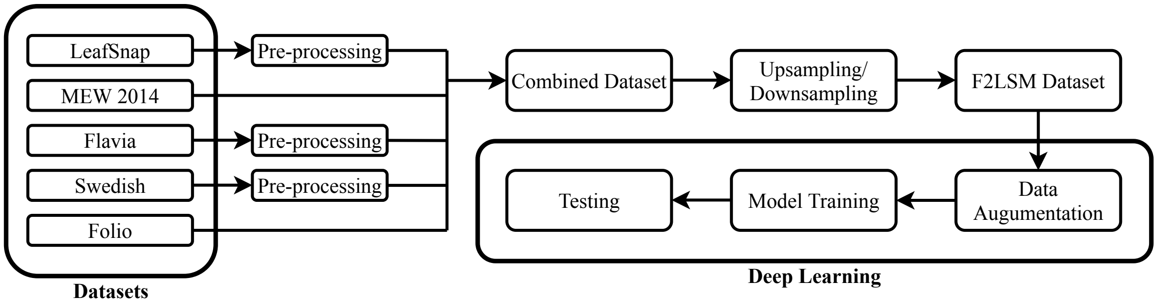


Figure 2. Flow Diagram of the procedure. Five publicly available datasets, LeafSnap, MEW 2014, Flavia, Swedish, and Folio, are combined into a single dataset after pre-processing. Resulting dataset is then balanced by upsampling the minority classes and downsampling the majority classes to form the F2LSM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

and 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]. Figure 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, 35, 36, 37, 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, 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.

## 2. RELATED WORK

Plant species identification based on leaf images has been an active area of research [8, 16, 17, 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%. Chaudhury 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 (DDLA) 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.

# 3. F2LSM DATASET

## 3.1. 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.

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 | |

**3.1.1. 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/`.

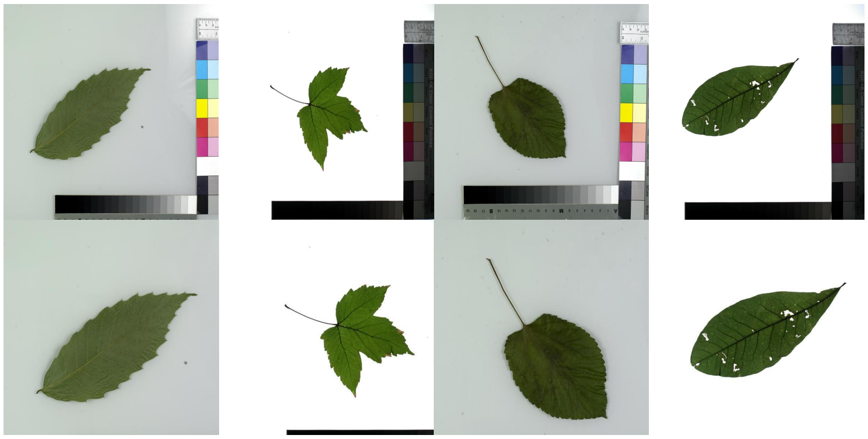


Figure 3. Class-specific cropping in the LeafSnap dataset.

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 Figure 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, Figure 4. We removed these 27 classes from the LeafSnap dataset.

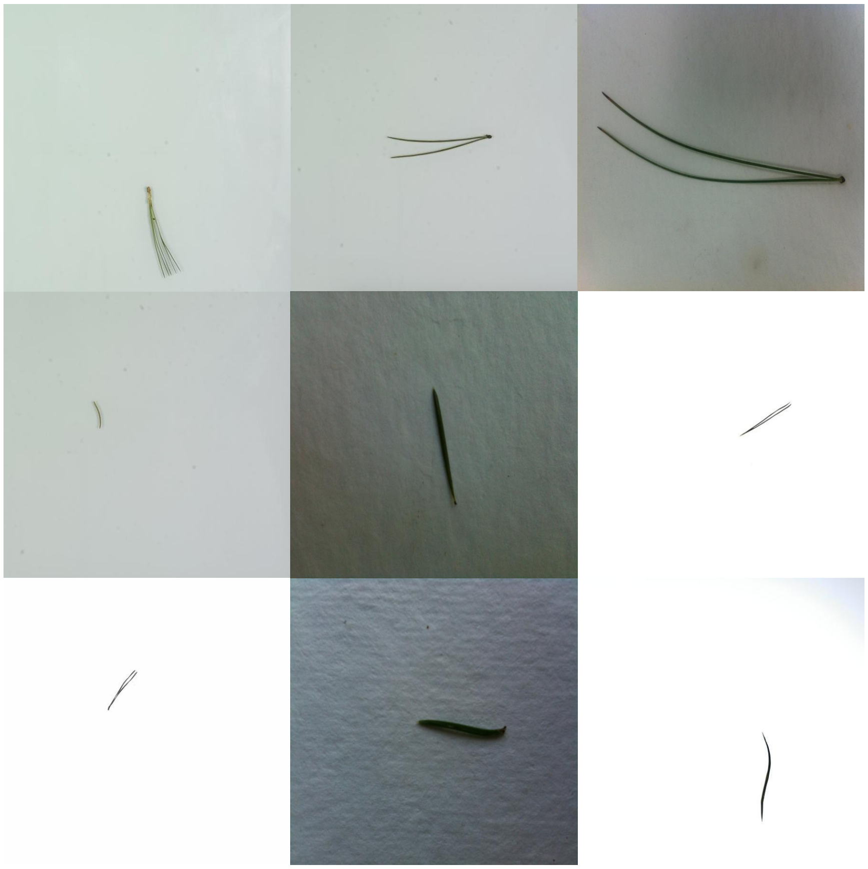


Figure 4. Image samples of different classes removed from the LeafSnap dataset, each image belongs to a different class.

**3.1.2. 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.

**3.1.3. 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 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.

**3.1.4. 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.

**3.1.5. 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.

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** |

## 3.2. 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 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:

**3.2.1. 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 $\sigma$ 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.

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

**3.2.2. 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.

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

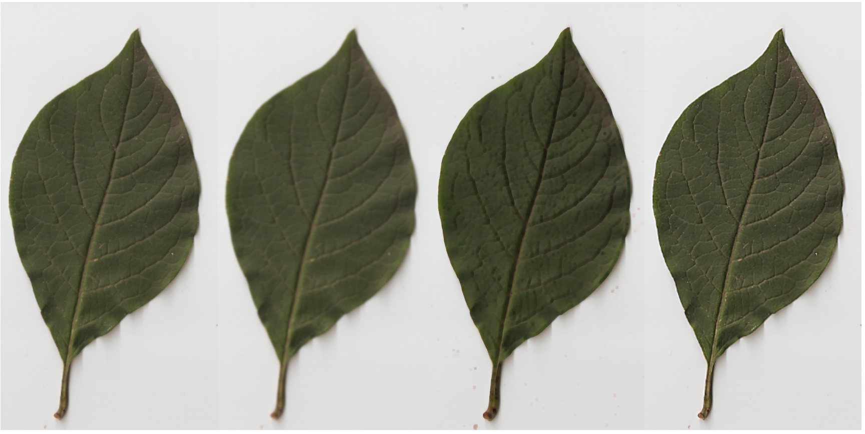


Figure 5. Different image transforms applied for oversampling classes with image samples less than 10. From left to right is the original image, a Gaussian blurred image, a rank filtered image, and an unsharp masked image.

**3.2.3. 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 $k = 1$ 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.

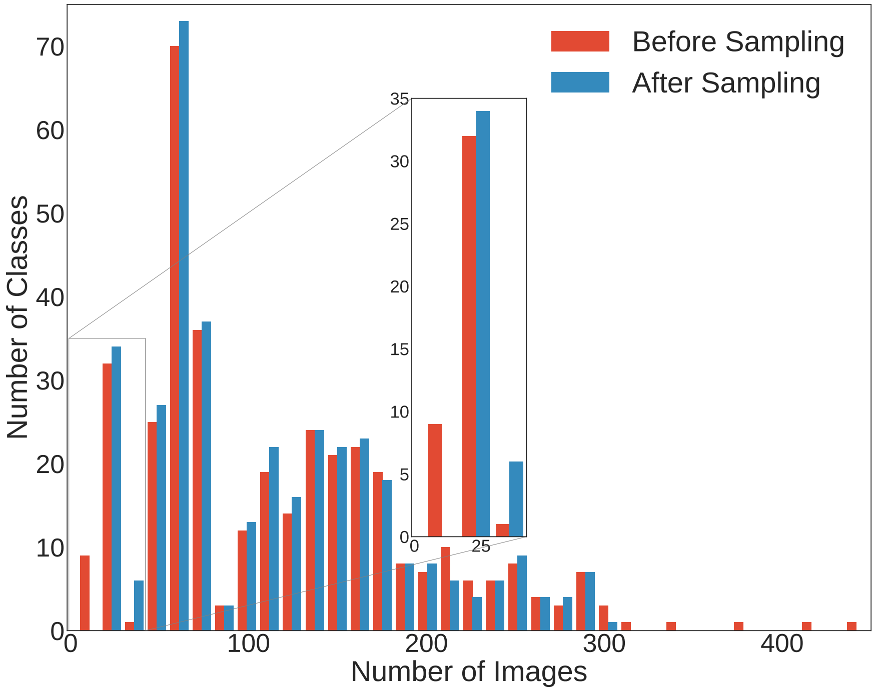


Figure 6. Histogram with 35 bins of the number of image samples per class before and after implementing oversampling and undersampling. The inset demonstrates that there were nine classes with less than ten images before sampling, and after sampling there are zero classes with less ten image samples.

Figure 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. Figure 6 shows the datasets' histogram before and after implementing oversampling and undersampling. The inset in Figure 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.

# 4. DEEP TRANSFER LEARNING

## 4.1. 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 Figure 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, 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.

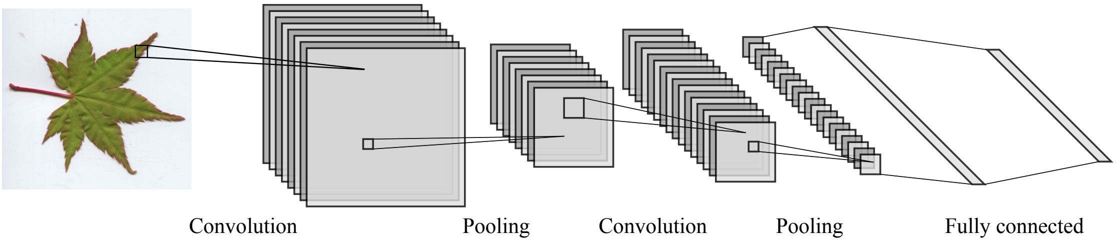


Figure 7. A typical CNN Architecture with multiple convolutional and pooling layers followed by fully connected layers for multi-class classification.

### 4.2. 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–EffcientNet [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 Figure 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.

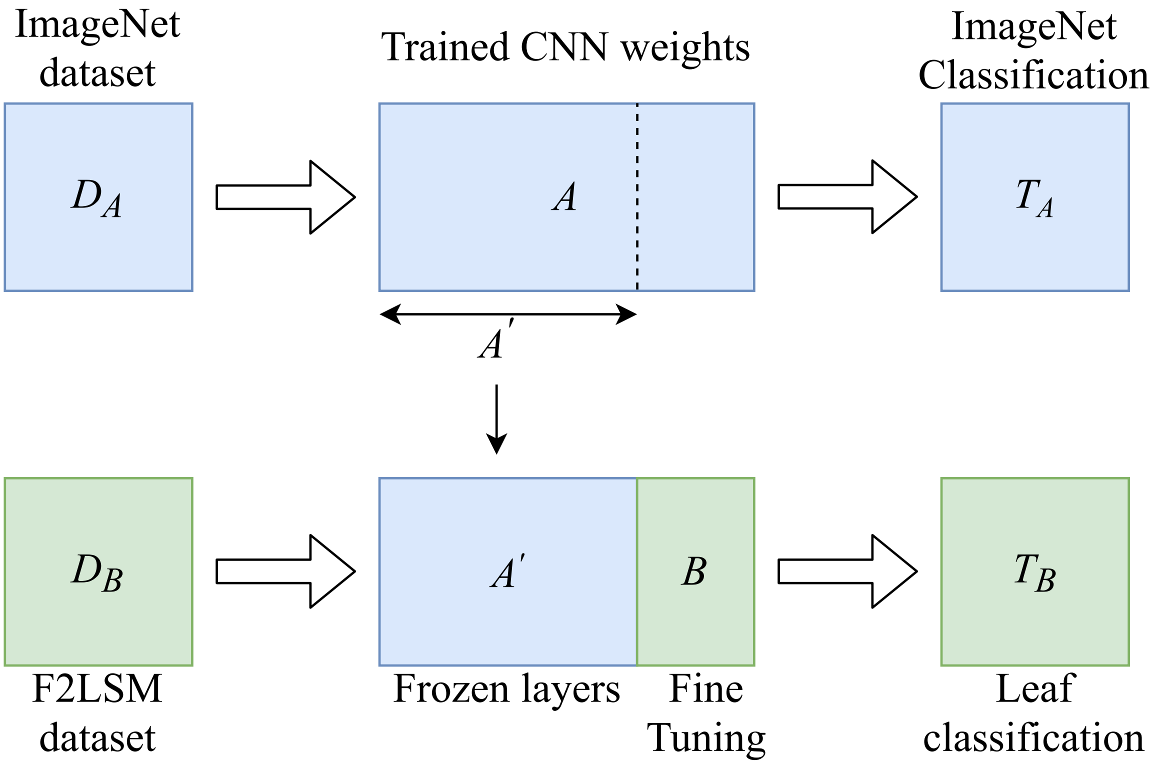


Figure 8. Transfer learning implemented on the F2LSM dataset. *A'* parameters learned for task A ($T_A$) are transferred to task B ($T_B$), and *B* parameters are fine-tuned.

# 5. EXPERIMENTS, RESULTS, AND ANALYSIS

## 5.1. 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 that the test set had a population that best represented the population

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] |

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.

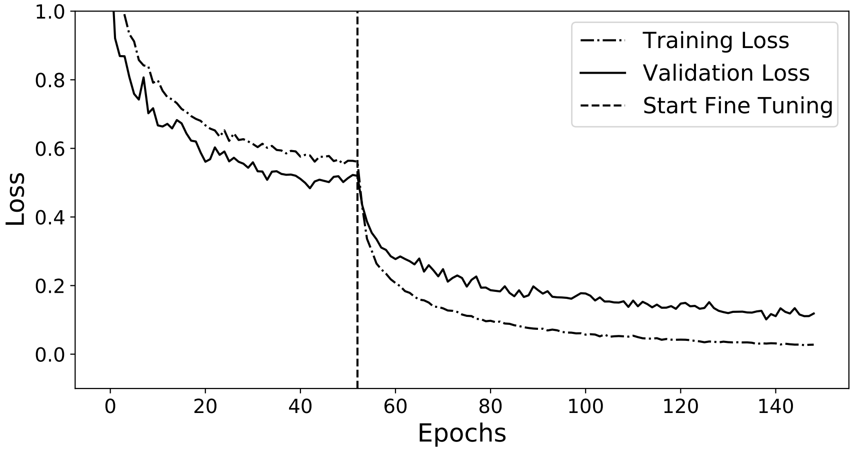


Figure 9. Loss trend while training EfficientNet B0, before and after fine-tuning the last few layers of the model.

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, Figure 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.

## 5.2. METRICS

The combined dataset is imbalanced, as shown in Figure 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:

$$Prec_l = \frac{TP_l}{TP_l + FP_l}, \tag{3}$$

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

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

Where $N_c$ is the total number of classes.

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

$$Rec_l = \frac{TP_l}{TP_l + FN_l} \tag{5}$$

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

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.

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

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

## 5.3. RESULTS AND DISCUSSION

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

**5.3.1. 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.

**5.3.2. 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 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

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** |

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 EffientNet. If computational complexity is a concern, then models such as MobileNet and EfficientNet B0 can be trained to use considerably fewer parameters and still provide high accuracy and F-scores. Figure 10 summarizes the number of parameters versus the F-score efficiency.

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** |

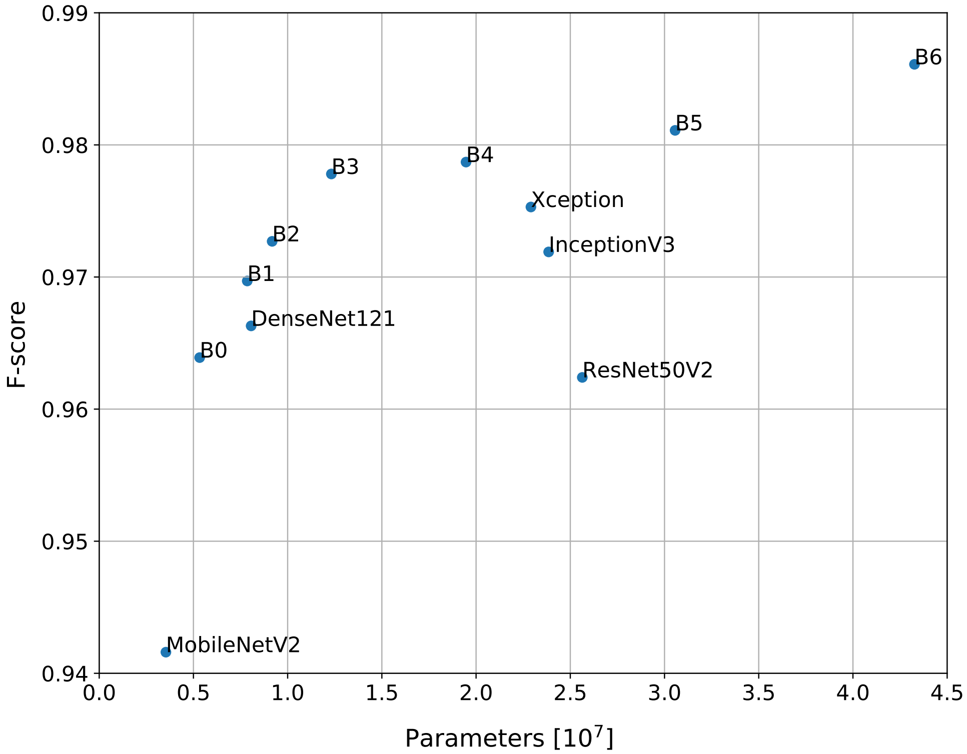


Figure 10. Number of model parameters vs. mean F-score. The plot shows that EfficientNet family of models perform better than other models for leaf identification in F2LSM dataset.

**5.3.3. 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]. Figure 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.

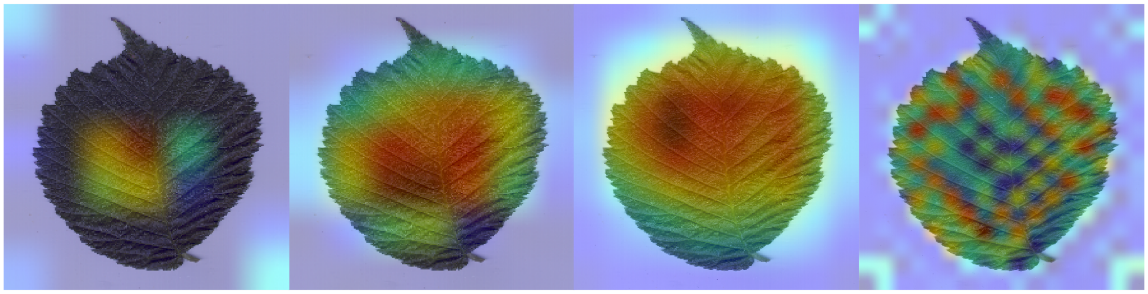


Figure 11. Grad-Cam visualizations of a leaf image for different networks. From left to right, MobileNetV2, EfficientNet B0, Xception, and EfficientNet B6, networks examine more detailed features as the number of parameters increase.

**5.3.4. Comparisons and Benchmarks.** The performance of the CNN architectures on the F2LSM dataset was bench-marked by comparing its performance with the accuracy obtained on the individual datasets used to create the new dataset, as shown in Table 6. It

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 [43] |
| Flavia [22] | 32 | Extreme learning machines [44] | 99.10 [44] |
| MEW 2012 [21] | 153 | SVM [45] | 96.54 [46] |
| LeafSnap [20] | 185 | Siamese CNN [47] | 96.00 [47] |
| F2LSM | 374 | EfficientNet B6 [15] | 98.41 |

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.

### 5.4. 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 [48], Tensorflow [49], and Keras [50] libraries for implementation of stratified k-fold cross-validation of CNN architectures.

## 6. 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.

## ACKNOWLEDGEMENTS

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. *Environmental Health Perspectives*, 109(suppl 1):69–75, 2001. doi: 10.1289/ehp.01109s169.

[2] Bradley C. Bennett and Michael J. Balick. Does the name really matter? the importance of botanical nomenclature and plant taxonomy in biomedical research. *Journal of Ethnopharmacology*, 152(3):387–392, 2014. ISSN 0378-8741. doi: https://doi.org/10.1016/j.jep.2013.11.042. URL `https://www.sciencedirect.com/science/article/pii/S0378874113008465`.

[3] Diego Rivera, Robert Allkin, Concepción Obón, Francisco Alcaraz, Rob Verpoorte, and Michael Heinrich. What is in a name? the need for accurate scientific nomenclature for plants. *Journal of Ethnopharmacology*, 152(3):393–402, 2014. ISSN 0378-8741. doi: https://doi.org/10.1016/j.jep.2013.12.022. URL `https://www.sciencedirect.com/science/article/pii/S0378874113009021`.

[4] Rainer W. Bussmann. Chapter 4 - Taxonomy–An irreplaceable tool for validation of herbal medicine. In Pulok K. Mukherjee, editor, *Evidence-Based Validation of Herbal Medicine*, pages 87–118. Elsevier, Boston, 2015. ISBN 978-0-12-800874-4. doi: https://doi.org/10.1016/B978-0-12-800874-4.00004-0. URL `https://www.sciencedirect.com/science/article/pii/B9780128008744000040`.

[5] Bradley C Bennett and Michael J Balick. Phytomedicine 101: plant taxonomy for preclinical and clinical medicinal plant researchers. *Journal of the Society for Integrative Oncology*, 6(4):150, 2008.

[6] Zhangnan Wu, Yajun Chen, Bo Zhao, Xiaobing Kang, and Yuanyuan Ding. Review of weed detection methods based on computer vision. *Sensors*, 21(11), 2021. ISSN 1424-8220. doi: 10.3390/s21113647. URL `https://www.mdpi.com/1424-8220/21/11/3647`.

[7] Zhenbo Li, Ruohao Guo, Meng Li, Yaru Chen, and Guangyao Li. A review of computer vision technologies for plant phenotyping. *Computers and Electronics in Agriculture*, 176:105672, 2020. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2020.105672. URL `https://www.sciencedirect.com/science/article/pii/S0168169920307511`.

[8] Jana Wäldchen and Patrick Mäder. Plant species identification using computer vision techniques: A systematic literature review. *Archives of Computational Methods in Engineering*, 25(2):507–543, 2018.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL `https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`.

[10] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 818–833, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10590-1.

[11] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[14] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4780–4789, Jul. 2019. doi: 10.1609/aaai.v33i01.33014780. URL `https://ojs.aaai.org/index.php/AAAI/article/view/4405`.

[15] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/tan19a.html`.

[16] Jana Waldchen, Michael Rzanny, Marco Seeland, and Patrick Mader. Automated plant species identification–trends and future directions. *PLOS Computational Biology*, 14 (4):1–19, 04 2018. doi: 10.1371/journal.pcbi.1005993. URL `https://doi.org/10.1371/journal.pcbi.1005993`.

[17] K Pankaja and G Thippeswamy. Survey on leaf recognization and classification. In *2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, pages 442–450, 2017. doi: 10.1109/ICIMIA.2017.7975654.

[18] O.H. Babatunde, L. Armstrong, J. Leng, and D. Diepeveen. A survey of computer-based vision systems for automatic identification of plant species. *Journal of Agricultural Informatics*, 6(1):61–71, 2015. doi: 10.17700/jai.2015.6.1.152. URL `https://researchrepository.murdoch.edu.au/id/eprint/25987/`.

[19] Anubha Pearline Sundara Sobitha Raj and Sathiesh Kumar Vajravelu. Ddla: dual deep learning architecture for classification of plant species. *IET Image Processing*, 13(12): 2176–2182, 2019.

[20] Neeraj Kumar, Peter N. Belhumeur, Arijit Biswas, David W. Jacobs, W. John Kress, Ida C. Lopez, and João V. B. Soares. Leafsnap: A computer vision system for automatic plant species identification. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 502–516, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-33709-3.

[21] Petr Novotný and Tomás Suk. Leaf recognition of woody species in central europe. *Biosystems Engineering*, 115(4):444–452, 2013. ISSN 1537-5110. doi: https://doi.org/10.1016/j.biosystemseng.2013.04.007. URL `https://www.sciencedirect.com/science/article/pii/S1537511013000731`.

[22] Stephen Gang Wu, Forrest Sheng Bao, Eric You Xu, Yu-Xuan Wang, Yi-Fan Chang, and Qiao-Liang Xiang. A leaf recognition algorithm for plant classification using probabilistic neural network. In *2007 IEEE International Symposium on Signal Processing and Information Technology*, pages 11–16, 2007. doi: 10.1109/ISSPIT.2007.4458016.

[23] Oskar Söderkvist. Computer vision classification of leaves from Swedish trees. In *MS Thesis*. Linköping University, Linköping, Sweden, 2001.

[24] Trishen Munisami, Mahess Ramsurn, Somveer Kishnah, and Sameerchand Pudaruth. Plant leaf recognition using shape features and colour histogram with k-nearest neighbour classifiers. *Procedia Computer Science*, 58:740–747, 2015. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2015.08.095. URL https://www.sciencedirect.com/science/article/pii/S1877050915022061. Second International Symposium on Computer Vision and the Internet (VisionNetâ15).

[25] Charles X. Ling and Chenghui Li. Data mining for direct marketing: Problems and solutions. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, KDD'98, page 73–79. AAAI Press, 1998.

[26] Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.*, 6(1):20–29, jun 2004. ISSN 1931-0145. doi: 10.1145/1007730.1007735. URL https://doi.org/10.1145/1007730.1007735.

[27] Alberto Fernández, Salvador Garcia, Francisco Herrera, and Nitesh V Chawla. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, 61:863–905, 2018.

[28] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In Věra Kůrková, Yannis Manolopoulos, Barbara Hammer, Lazaros Iliadis, and Ilias Maglogiannis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2018*, pages 270–279, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01424-7.

[29] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[30] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[31] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[32] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[33] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[34] Ayan Chaudhury and John L. Barron. Plant species identification from occluded leaf images. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17 (3):1042–1055, 2020. doi: 10.1109/TCBB.2018.2873611.

[35] Munish Kumar, Surbhi Gupta, Xiao-Zhi Gao, and Amitoj Singh. Plant species recognition using morphological features and adaptive boosting methodology. *IEEE Access*, 7:163912–163918, 2019. doi: 10.1109/ACCESS.2019.2952176.

[36] Jiang Huixian. The analysis of plants image recognition based on deep learning and artificial neural network. *IEEE Access*, 8:68828–68841, 2020. doi: 10.1109/ACCESS.2020.2986946.

[37] Xuan Wang, Weikang Du, Fangxia Guo, and Simin Hu. Leaf recognition based on elliptical half gabor and maximum gap local line direction pattern. *IEEE Access*, 8: 39175–39183, 2020. doi: 10.1109/ACCESS.2020.2976117.

[38] Yupeng Song, Fazhi He, and Xiying Zhang. To identify tree species with highly similar leaves based on a novel attention mechanism for cnn. *IEEE Access*, 7:163277–163286, 2019. doi: 10.1109/ACCESS.2019.2951607.

[39] Viraj Gajjar, Ze-Hao Lai, and Kurt Kosbar. Fast classification of leaf images for agricultural remote sensing applications. In *International Telemetering Conference*, volume 54, pages 569–578, 2018.

[40] Jing wei Tan, Siow-Wee Chang, Sameem Abdul-Kareem, Hwa Jen Yap, and Kien-Thai Yong. Deep learning for plant species classification using leaf vein morphometric. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(1):82–90, 2020. doi: 10.1109/TCBB.2018.2848653.

[41] Xinchuan Zeng and Tony R. Martinez. Distribution-balanced stratified cross-validation for accuracy estimation. *Journal of Experimental & Theoretical Artificial Intelligence*, 12(1):1–12, 2000. doi: 10.1080/095281300146272. URL `https://doi.org/10.1080/095281300146272`.

[42] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[43] Pornntiwa Pawara, Emmanuel Okafor, Lambert Schomaker, and Marco Wiering. Data augmentation for plant classification. In Jacques Blanc-Talon, Rudi Penne, Wilfried Philips, Dan Popescu, and Paul Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, pages 615–626, Cham, 2017. Springer International Publishing. ISBN 978-3-319-70353-4.

[44] Muammer Turkoglu and Davut Hanbay. Recognition of plant leaves: An approach with hybrid features produced by dividing leaf images into two and four parts. *Applied Mathematics and Computation*, 352:1–14, 2019. ISSN 0096-3003. doi: https://doi.org/10.1016/j.amc.2019.01.054. URL https://www.sciencedirect.com/science/article/pii/S0096300319300694.

[45] E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In *Neural Networks for Signal Processing VII. Proceedings of the 1997 IEEE Signal Processing Society Workshop*, pages 276–285, 1997. doi: 10.1109/NNSP.1997.622408.

[46] Christian Reul, Martin Toepfer, and Frank Puppe. Cross dataset evaluation of feature extraction techniques for leaf classification. *International Journal of Artificial Intelligence Applications*, 7:01–20, 03 2016. doi: 10.5121/ijaia.2016.7201.

[47] Voncarlos M. Araújo, Alceu S. Britto Jr., Luiz S. Oliveira, and Alessandro L. Koerich. Two-view fine-grained classification of plant species. *Neurocomputing*, 467:427–441, 2022. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2021.10.015. URL https://www.sciencedirect.com/science/article/pii/S0925231221014934.

[48] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010. doi: 10.25080/Majora-92bf1922-00a.

[49] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[50] Francois Chollet et al. Keras, 2015. URL https://github.com/fchollet/keras.

# III. CSI ESTIMATION USING ARTIFICIAL NEURAL NETWORK

Viraj K. Gajjar, and Kurt L. Kosbar
Department of Electrical Engineering
Missouri University of Science and Technology
Rolla, Missouri 65409–0050
Tel: 573–202–1062
Email: vgf4c@mst.edu

## ABSTRACT

We propose using machine learning to estimate channel state information (CSI) for MIMO communication links. The goal is to use information such as atmospheric conditions, amount of path loss, and Doppler shift to improve the accuracy of CSI estimates. We start by designing an algorithm which estimates the CSI based on previously mentioned factors. Using this algorithm, we simulate a dataset of CSI over varying atmospheric conditions, receiver position, and receiver velocity. We then use this dataset to train an artificial neural network, which is able to estimate the CSI by using the current atmospheric condition, receiver position, and velocity.

**Keywords:** channel state information, artificial neural network, mimo systems

## 1. INTRODUCTION

In wireless communications, CSI describes how a signal propagates through the medium. Accuracy of the CSI estimates is one of the major factors which limits bit-error-rate (BER) and choice of modulation schemes. Techniques that can accurately estimate CSI are essential for many high-performance systems. CSI incorporates the effects of large-scale path loss, small-scale fading, and multipath [1], which on more fundamental level, depends on factors such as the distance between the transmitter and the receiver, weather conditions, velocity of the transmitter and receiver, carrier frequency, size of the scattering objects, type of reflecting surfaces. This paper investigates how a function can be obtained, that maps the factors affecting the wireless channel and the CSI using machine learning tools.

The recent growth in computational power has provided an opportunity to process large amounts of data, and learn complex relations from it, using machine learning (ML) algorithms. A class of ML algorithms known as supervised learning have the ability to learn a function that maps an input to the output based on available data, which consists of input-output pairs [2]. This method has been applied in the wireless communications to estimate channel noise statistics in multiple-inputmultiple-output (MIMO) wireless networks [3]. In another application, it learned a mobile terminal's usage pattern in a cellular system to provide optimal handover solutions [4]. Other applications include improvement in pilot contamination in cellular massive MIMO by learning channel parameters of the desired link in the target cell and undesired link in the adjacent cell [5], and to estimate CSI for 5G cellular communications using convolutional neural networks [6]. Artificial neural networks (ANNs) [7] are useful in solving problems related to function fitting, classification, time series prediction, and clustering. In regards to function fitting problems, ANNs have the ability to approximate any continuous function under certain assumptions [8]. Thus, machine learning (ML) techniques such as ANNs offer a possibility to find a complex function that can map the factors affecting the wireless channel to its CSI.

This paper presents an ANN based CSI estimation technique which uses simulated data to train an ANN for CSI estimation. Specifically, CSI data is simulated for a MIMO network across varying atmospheric conditions, and varying receiver position and velocity. Approximately two-thirds of the simulated data is used to train an ANN, which uses the distance between the transmitter and the receiver, velocity of the receiver, and the atmospheric conditions as features. The remainder of the data is used to test and validate the efficiency of the trained network.

# 2. THE DATASET

For simulating the dataset, certain parameters that affect the propagation of a signal were considered, and are as follows:

## 2.1. FREE SPACE PATH LOSS

Free space path loss (PL) represents signal attenuation in dB for line-of-sight communications, and can be calculated using the following equation [1]:

$$PL(dB) = -10log\left[\frac{G_tG_r\lambda^2}{(4\pi)^2d^2}\right], \tag{1}$$

where $G_t$ and $G_r$ are the transmitter and the receiver antenna gain respectively, $\lambda$ is the wavelength of the signal, and $d$ is the distance between the transmitting and receiving antenna.

## 2.2. DOPPLER SHIFT

When the receiver is moving with respect to the transmitter, there will be an apparent change in frequency receiver known as Doppler shift $f_d$.The Doppler shift can be calculated using the following equation [1]:

$$f_d = \frac{v}{\lambda}cos\theta, \tag{2}$$

where $v$ is the velocity of the receiver, $\lambda$ is the wavelength, and $\theta$ is the angle of elevation from the receiver to the source. The Doppler shift is positive if the receiver is moving towards the transmitter and it is negative if the receiver is moving away from the transmitter.

## 2.3. RAINFALL ATTENUATION

The attenuation of electromagnetic signals due to rain for frequencies up to 1,000 GHz can be approximated using the following equation [9]:

$$\gamma_r = kR^{\alpha}, \tag{3}$$

where $\gamma_r(dB/km)$ is the specific attenuation based on rain-rate $R(mm/hr)$, and coefficients $k$ and $\alpha$ depend on the frequency, the polarization state, and the elevation angle of the of the signal path.

## 2.4. ATMOSPHERIC GAS ATTENUATION

The attenuation of signal at frequencies up to 1,000 GHz due to dry air and water vapor, can be evaluated for given pressure, temperature and humidity using the following model [10]:

$$\gamma = \gamma_o + \gamma_w = 0.1820fN''(f), \tag{4}$$

where $\gamma(dB/km)$ is the attenuation due to gases, $\gamma_o$ and $\gamma_w$ are the specific attenuations in $(dB/km)$ due to dry air (oxygen, pressure-induced nitrogen, and non-resonant Debye attenuation) and water vapor, respectively, $f(GHz)$ is the frequency, and $N''(f)$ is the imaginary part of the complex atmospheric refractivity [10].

## 2.5. FOG AND CLOUD ATTENUATION

Attenuation within a cloud or fog is given as [11]:

$$\gamma_c(f,T) = K_l(f,T)M, \tag{5}$$

where $\gamma_c(dB/km)$ is the attenuation within the cloud, $k_l((dB/km)/(g/m^3))$ cloud liquid water attenuation coefficient, $M(g/m^3)$ is the liquid water density, $f(GHz)$ is the frequency, and $T(K)$ is the cloud liquid water temperature.

## 2.6. DATASET GENERATION

A dataset of 1.5 million samples was generated considering these attenuation models for a 2-by-2 MIMO system as shown in Figure 1, using the following equation:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{21} \\ h_{12} & h_{22} \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \tag{6}$$

where $p_1$ and $p_2$ are the pilot signals transmitted from transmitter-1 and transmitter-2 respectively; $h_{ij}$ is the channel coefficient from $i^{th}$ transmitter to $j^{th}$ receiver, where $i = 1, 2$, and $j = 1, 2$; $y_1$ and $y_2$ are the received signals at receiver-1 and receiver-2 respectively after being subjected to the attenuation models discussed earlier. In order to cover a range of possibilities, the parameters that affect the CSI were varied as stated in Table 1.

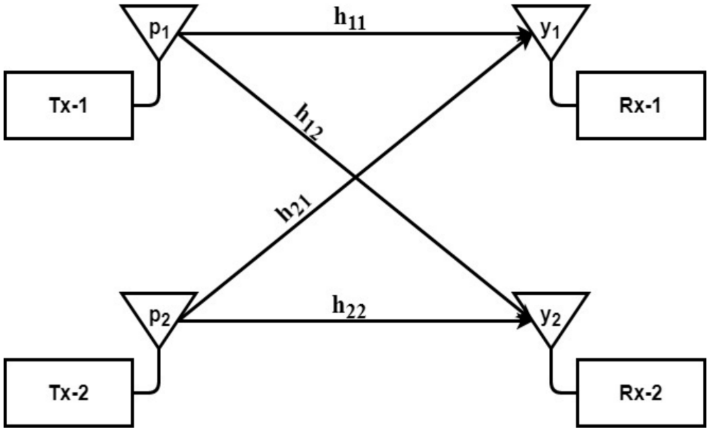


Figure 1. A 2-by-2 MIMO system for simulating the dataset.

Table 1. Dataset parameter range.

| Parameter | Range |
|---|---|
| Distance between transmitter and receiver | 1 km to 3 km |
| Velocity of the receivers | -10 (m/s) to 10 (m/s) |
| Temperature | $-30°$ C to $30°$ C |
| Carrier frequency | 1 MHz to 30 MHz |
| Rain-rate | 0 (mm/hr) to 50 (mm/hr) |
| Fog | 0.05 (g/m$^3$) to 0.5 (g/m$^3$) |
| Humidity | changed with respect to the temperature [12] |

## 3. ARTIFICIAL NEURAL NETWORK

The dataset consists of two matrices, an input feature matrix and an output CSI matrix. The input feature matrix consists of 1.5 million samples of the eight following features: carrier frequency, the distance between the transmitter and receiver, velocity of receiver, angle of elevation/depression from transmitting antenna to the receiving antenna, temperature, rain-rate, fog-density, and humidity. The output CSI matrix also has 1.5 million samples of CSI, where the four channel coefficients of each sample are split into their respective real and imaginary parts; therefore, each sample in the CSI matrix is a vector of length eight. Thus, the neural network should have eight neurons at the input which takes a vector of features as input, and eight neurons at the output which estimates real and imaginary part of four channel coefficients.

Before training the ANN, the input feature matrix is normalized, and both the input and the output datasets are divided into a training set, a validation set, and a test set. The output dataset was scaled up by a factor of $10^4$, as the values of the channel coefficients were in the range of $10^{-2}$ – $10^{-10}$. Approximately 70% samples of the entire dataset and is used to train the ANN. The validation set forms 15% of the dataset, and is used to provide an unbiased evaluation of the training data fit while tuning model parameters. The remaining 15% of the dataset is the test set, which is used to evaluate the final model.

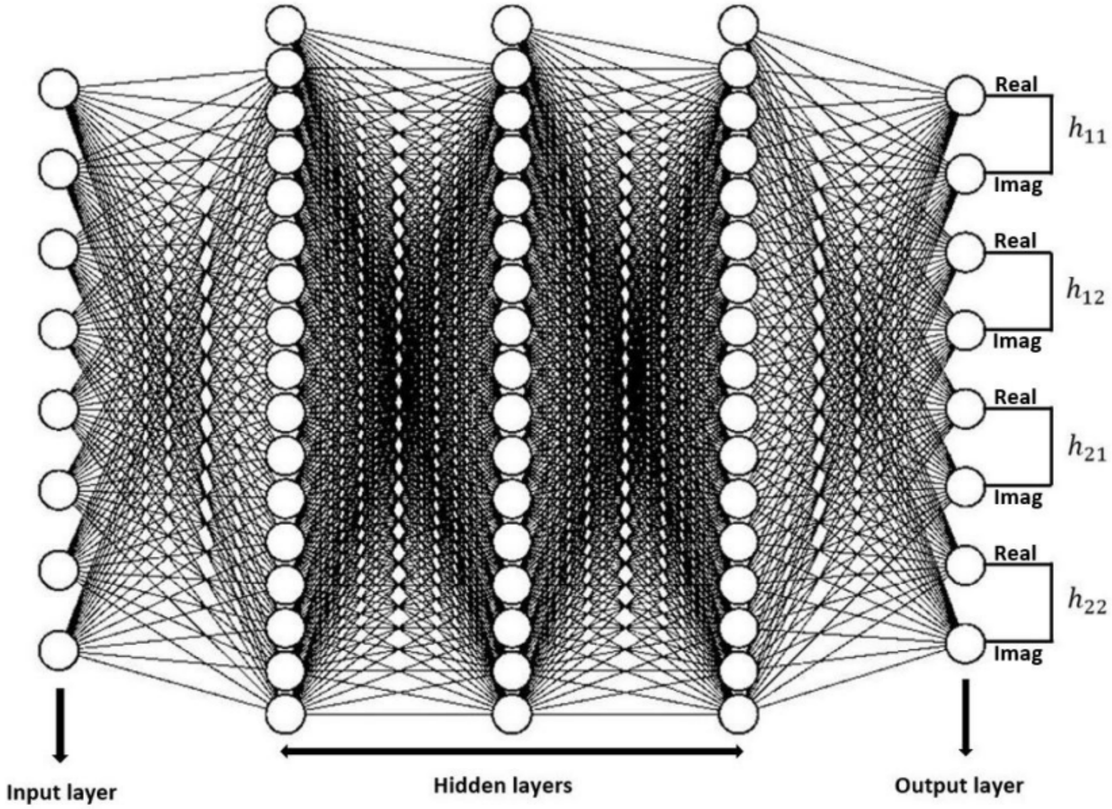


Figure 2. ANN with an input layer, three hidden layers, and an output layer.

The overall architecture of the ANN trained is shown in Figure 2. The network consists of five layers - the input layer and the output layer with eight neurons each, and three hidden layers with 17 neurons each including one bias neuron.

All the weights (connections between each neuron) are randomly initialized before the training process starts. The training process starts by feeding a feature vector to the input layer and based on this input the activations of the next layer are calculated. The network calculates the activations of each subsequent layer using the following equation:

$$A^{(1)} = \sigma(WA^{(0)} + b) \tag{7}$$

where $A^{(1)}$ and $A^{(0)}$ are activations of the output layer and the input layer respectively, $W$ is the matrix of weights connecting the input and the subsequent layer, $b$ is the bias vector, and $\sigma$ is the activation function. (8) and (9) are the activation functions used for the hidden layers and the output layer respectively. The linear function (9) at the output layer is better at fitting a function compared to the tan-sigmoid function (8).

After calculating the activations of each neuron, the cost function of the entire network is calculated by comparing the network outputs and actual outputs. This cost function is then used to calculate the gradient using Levenberg-Marquardt backpropagation [13], then the calculated gradient is used to perform gradient descent to update the weights. The network iterates through the entire dataset several times until either expected accuracy is achieved or the performance plateaus.

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{8}$$

$$f(x) = x \tag{9}$$

## 4. RESULTS AND DISCUSSION

The ANN was trained for more than 8,000 epochs, and then the network was tested on the test set which consisted of completely new samples which the network wasn't exposed to during the training stage. Before analyzing the performance of the algorithm, it should be noted that the outputs range in the order of $10^2$ to $10^{-6}$. Figure 3 shows the Mean Squared Error (MSE) of the entire training set over several epochs.

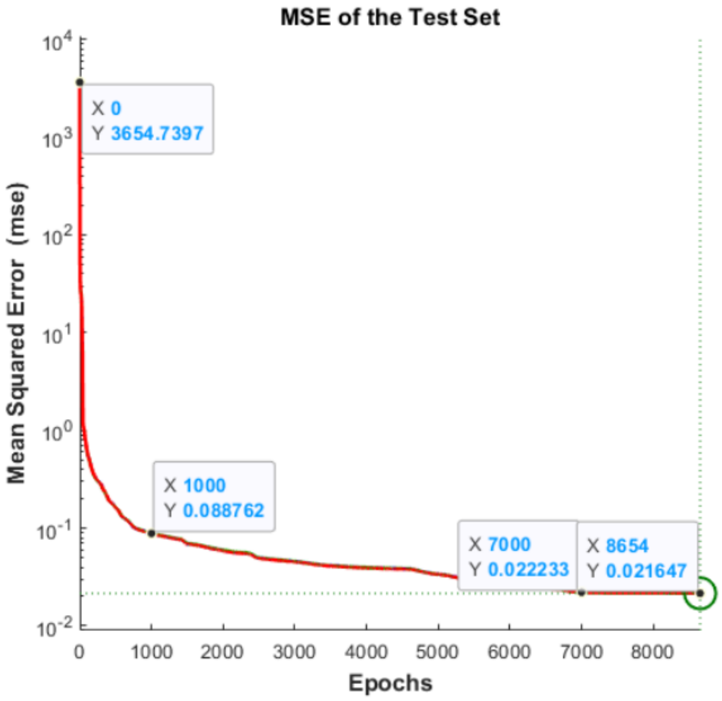


Figure 3. MSE of the test set plotted against the number of epochs.

Before the training starts, the performance of the network is poor with the MSE being approximately 3,650. But once the network starts training the performance improves rapidly till 1,000 epochs, and the MSE drops to approximately 0.09.

However, the improvement in the performance isn't substantial after 1,000 epochs, and it only drops by roughly 0.06 over 6,000 iterations. After that point, the improvement in the performance is negligible. The final performance of approximately 0.02 over the entire test set shows the ANN was able to closely fit a function that is able to map the input features to the CSI.

Figure 4 shows the error histogram with 20 bins, and illustrates how far the outputs deviate from the target values. As the histogram suggests, the majority of the samples are close to the zero error for all the training, validation and test sets. The spread of the histogram, just like MSE, narrowed rapidly till 1,000 epochs, but then the error decreased gradually for the next 6,000 iterations.

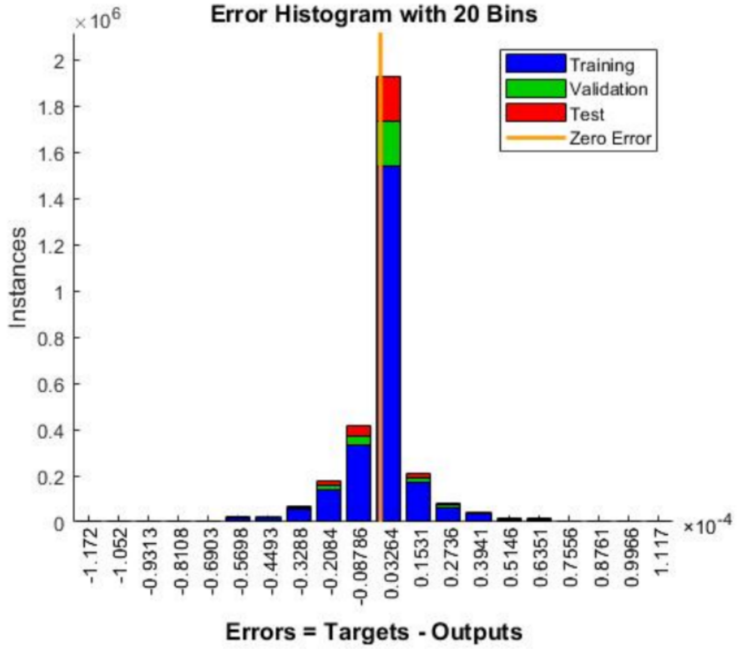


Figure 4. Error histogram.

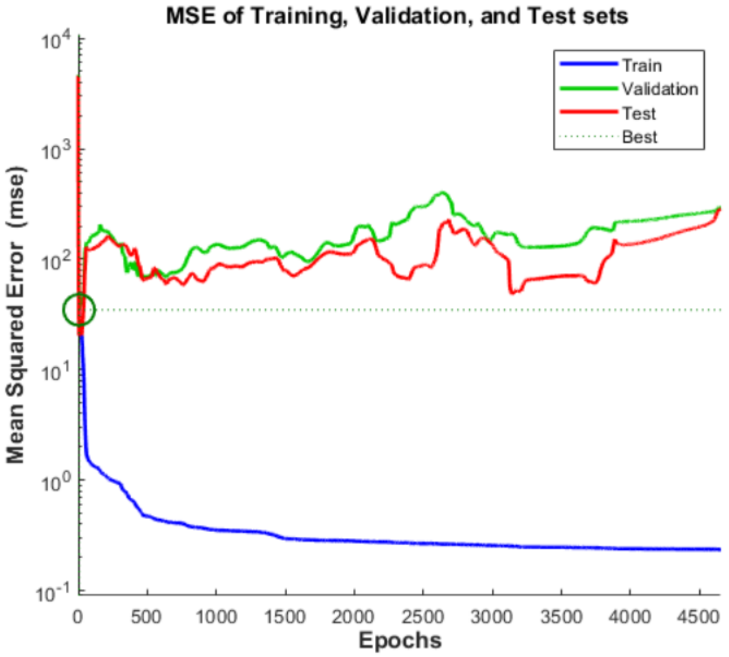


Figure 5. MSE with a small dataset demonstrates the effect of overfitting.

The amount of data used to train the ANN considerably affected the performance of the algorithm. The accuracy of the system increased with the size of the dataset till one million samples, but after that, there was only a marginal improvement in the performance. With smaller datasets the performance of the network was perfect for the training set, but the network struggled to perform on the test dataset. This meant the network did not have enough data to train, and was overfitting. Figure 5 demonstrates the effect of overfitting, as it can be seen that the neural network is able to fit a function across the test set, but as the function is specific to the training samples it fails to recognize test and validation samples.

The ANN was successful in learning a function that mapped the features to CSI. This was expected for a simulated dataset as the CSI was generated using several functions defining the amount of path loss, weather conditions, and the amount of Doppler shift. Therefore, the ANN was able to come up with a function that incorporated the effect of all the functions used to generate the dataset.

## 5. CONCLUSIONS

Our simulations suggest that the relation between the CSI of a MIMO communication link and the parameters affecting it can be exploited using machine learning tools. We demonstrated that an ANN was able to fit a function that mapped features affecting CSI - atmospheric conditions, the distance between the transmitters and receivers, the amount of Doppler shift - to its estimates. Although the ANN was only tested on the simulated dataset, with sufficient data and a deep enough network, this method may be useful for real-world applications.

## REFERENCES

[1] Theodore S Rappaport et al. *Wireless communications: principles and practice*, volume 2. prentice hall PTR New Jersey, 1996.

[2] Stuart Russel, Peter Norvig, et al. *Artificial intelligence: a modern approach*. Pearson Education Limited London, 2013.

[3] Vin-sen Feng and Shih Yu Chang. Determination of wireless networks parameters through parallel hierarchical support vector machines. *IEEE Transactions on Parallel and Distributed Systems*, 23(3):505–512, 2012. doi: 10.1109/TPDS.2011.156.

[4] Brad K. Donohoo, Chris Ohlsen, Sudeep Pasricha, Yi Xiang, and Charles Anderson. Context-aware energy enhancements for smart mobile devices. *IEEE Transactions on Mobile Computing*, 13(8):1720–1732, 2014. doi: 10.1109/TMC.2013.94.

[5] Chao-Kai Wen, Shi Jin, Kai-Kit Wong, Jung-Chieh Chen, and Pangan Ting. Channel estimation for massive mimo using gaussian-mixture bayesian learning. *IEEE Transactions on Wireless Communications*, 14(3):1356–1368, 2015. doi: 10.1109/TWC.2014.2365813.

[6] Changqing Luo, Jinlong Ji, Qianlong Wang, Xuhui Chen, and Pan Li. Channel state information prediction for 5G wireless communications: A deep learning approach. *IEEE Transactions on Network Science and Engineering*, 7(1):227–236, 2020. doi: 10.1109/TNSE.2018.2848960.

[7] Donald F Specht et al. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[8] Balázs Csanád Csáji et al. Approximation with artificial neural networks. In *MS Thesis*. Faculty of Sciences, Etvs Lornd University, Hungary, 2001.

[9] ITU Radiocommunication Sector. P.838 : Specific attenuation model for rain for use in prediction methods. `https://www.itu.int/rec/R-REC-P.838/en`, 2005.

[10] ITU Radiocommunication Sector. P.676 : Attenuation by atmospheric gases. `https://www.itu.int/rec/R-REC-P.676`, 2013.

[11] ITU Radiocommunication Sector. P.840 : Attenuation due to clouds and fog. `https://www.itu.int/rec/R-REC-P.840`, 2013.

[12] Transport Information Service. Climate/humidity table. `https://www.tis-gdv.de/tis_e/misc/klima.htm/`.

[13] G. Lera and M. Pinzolas. Neighborhood based levenberg-marquardt algorithm for neural network training. *IEEE Transactions on Neural Networks*, 13(5):1200–1203, 2002. doi: 10.1109/TNN.2002.1031951.

# IV. RAPID GAIN ESTIMATION FOR MULTI-USER SOFTWARE DEFINED RADIO APPLICATIONS

Viraj K. Gajjar, and Kurt L. Kosbar
Department of Electrical Engineering
Missouri University of Science and Technology
Rolla, Missouri 65409–0050
Tel: 573–202–1062
Email: vgf4c@mst.edu

## ABSTRACT

This paper proposes a machine learning algorithm to estimate the peak of a signal generated as a sum of modulated signals. Each signal in the sum may have different modulation format, carrier frequency, data rate, and power level. A dataset of summed signals with varying parameters of individual signals was simulated to train an artificial neural network. The neural network estimates the peak voltage, central moments, variance, skewness, and kurtosis of the summed signal. Once trained, the neural network can rapidly and accurately predict the peak voltage, and statistics of the summed signal, based on the individual signal parameters, and does not have to generate or observe the signal itself. This can be used in an automatic gain control system, to prevent clipping when multiple modulators share a digital to analog converter or amplifier chain.

**Keywords:** software defined radio, digital to analog converter, gain estimation, statistical moments, artificial neural networks

## 1. INTRODUCTION

A software defined radio (SDR) can be used to generate multiple signals with varying modulation formats, data rates, power levels, and carrier frequency [1]. The multiple baseband modulated signals can be added in the SDR, and the summed signal sent to a digital-to-analog converter (DAC) for transmission. To minimize quantization noise, the summed signal needs to be as large as possible. However, exceeding the range of DAC will cause distortion due to clipping. It is possible to look for the peak voltage during transmission, and then adjust the gain based on the peak voltage. But using such a system may lead to distortion in first part of the transmission, and the problem gets worse if the number of transmitters allotted to the SDR changes. This paper proposes a way to estimate gain based on signal parameters such as modulation formats, data rates, power levels, and carrier frequencies of the summed signals and the number of signals added; without having to look at the summed signal.

Estimating statistics of the summed signal such as mean, variance, and standardized moments [2] can help determine how frequently any particular peak voltage occurs in the summed signal. If the peak is an outlier, then the system may consider clipping it for better efficiency, and adjust the gain based on some smaller peak which occurs more frequently. Estimating the peak voltage and the statistics of the summed signal based solely on the signal parameters can be considered as a function approximation problem.

A class of machine learning algorithms, supervised learning, can learn function that maps an input to the output based on available data, which consists of input-output pairs [3]. Supervised learning has been used in applications such as estimating the channel state information for a MIMO communication link based on atmospheric conditions [4], to estimate channel noise statistics in MIMO wireless network [5], to provide optimal handover solutions by learning a mobile terminal's usage pattern [6], and to improve pilot contamination in massive MIMO by learning channel parameters of the desired link in target cell and undesired link in the adjacent cells [7]. Supervised machine learning algorithms such as artificial neural networks (ANN) can solve function fitting problems, and can approximate nearly any continuous function under certain assumptions [8, 9].

This paper demonstrates a way to use ANN for estimating peak voltage and statistics of a signal for gain estimation using simulated data. Specifically, multiple signals with varying modulation formats, power levels, data rates, and carrier frequency are summed, and the peak voltage and the statistics of this summed signal are measured. The parameters of the signals summed, and the number of signals, form the input of the ANN, and the peak voltage and the statistics form the output. A dataset using 125,000 summed signals is generated, of which 70% of the data is used to train, and the remaining 30% is used to validate and test the efficiency of the trained ANN.

## 2. THE DATASET

The summed signal is generated by adding multiple signals with varying modulation formats, data rates, power levels, and carrier frequencies. Table 1 shows the range of parameters used to generate the summed signals for the dataset.

Table 1. Dataset parameter range

| **Parameter** | **Range** |
|---|---|
| Modulation technique | BPSK, QPSK, 8-QAM, 16-QAM, 32-QAM, 64-QAM, 128-QAM, and 256-QAM [10] |
| Bit rate | 1000 bps to 10000 bps |
| Power level | -30 dBm to 0 dBm |
| Normalized carrier frequency | -60 kHz to 60 kHz |

Figure 1 shows the power spectrum of the summed signal, where ten signals are added. To generate a summed signal, multiple signals are generated by randomly varying the parameters within the range of each parameter mentioned in Table 1. The following statistics of the summed signal were considered for simulating the dataset and for estimation: mean, variance, skewness (third standardized moment), kurtosis (fourth standardized moment), fifth standardized moment, and sixth standardized moment. Standardized moments can be found using (1).

$$\zeta_k = \frac{\mu_k}{\sigma^k} = \frac{E[(X-\mu)^k]}{\left(\sqrt{E[(X-\mu)^2]}\right)^k} \tag{1}$$

where, $\zeta$ is the $k^{th}$ standardized moment, $\mu_k$ is the $k^{th}$ moment about the mean, $\sigma$ is the standard deviation, $X$ is a random variable, and $\mu$ is the mean. Figure 2 shows how the statistics change as more samples of the summed signal are considered. The mean converges more rapidly than any other statistic. As the order of the moments increase, more samples of the signal are required for the statistic to converge. These statistics are calculated for a summed signal with 10 modulated signals, which is the maximum number of signals considered for dataset simulation. Figure 3 shows that peak voltage takes the largest number of samples to converge when compared to the statistics.

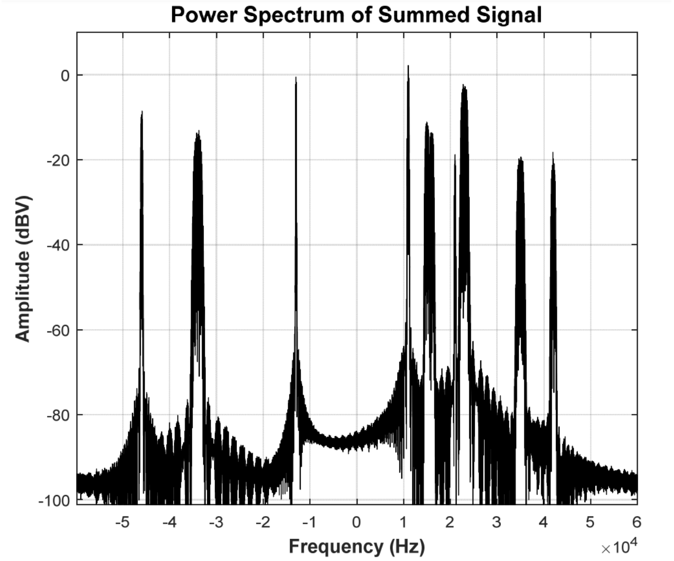


Figure 1. Power spectrum of the summed signal.

A dataset is generated using 125000 summed signals, where each summed signal is generated by randomly selecting parameters from Table 1. The number of modulated signals used to generate a summed signal can vary between 2 and 10.

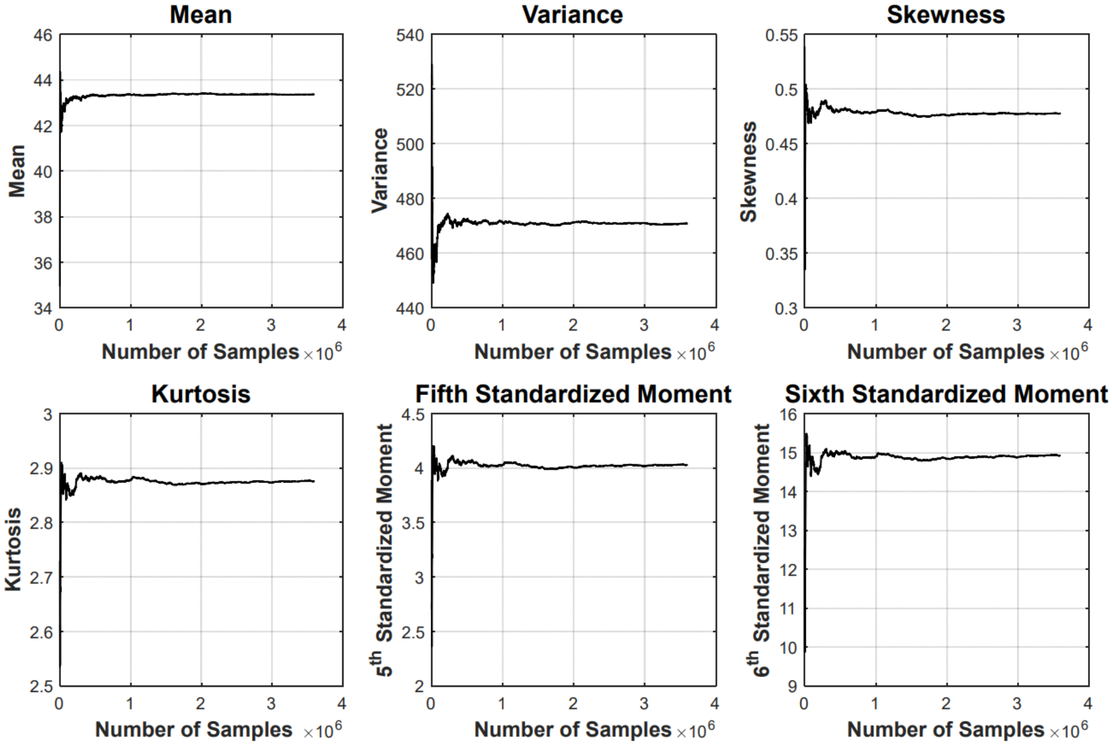


Figure 2. Convergence of statistics as the number of samples increase.

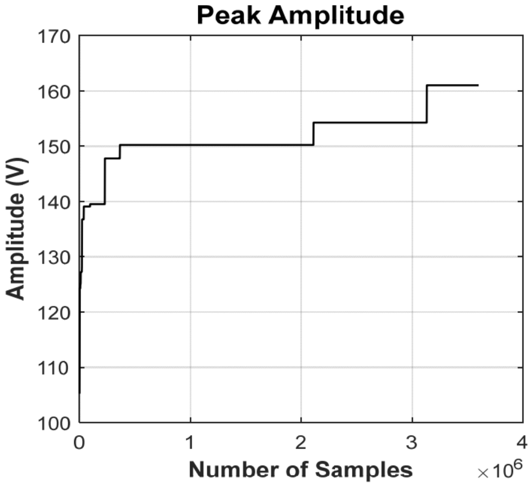


Figure 3. Convergence of peak amplitude as the number of samples increase.

## 3. ARTIFICIAL NEURAL NETWORK

The generated dataset consists of two matrices, an input feature matrix, and an output matrix of statistics and peak amplitudes. The input feature matrix consists of parameters: modulation order, bitrate, power level, and carrier frequency. The parameters are specified for each component signal used to generate the summed signal. The number of signals in the sum is also part of the feature matrix. The output matrix consists of the six statistics and the peak amplitude of the summed signal. Therefore, the input feature matrix is $125,000$-by-41, where $125,000$ are the number of data samples and 41 represents the number features for each data sample (4 features for each component signal for a maximum of 10 added signals, and one feature for the number of signals added). If fewer than ten signals are summed, the input features for unused signals are set to zero. Similarly, the output feature matrix is $125,000$-by-7, where 7 represents the peak amplitude and the six different statistics. Thus, the neural network should have 41 neurons at the input which takes a vector of features as input, and 7 neurons at the output which estimates the peak voltage and the statistics of the summed signal.

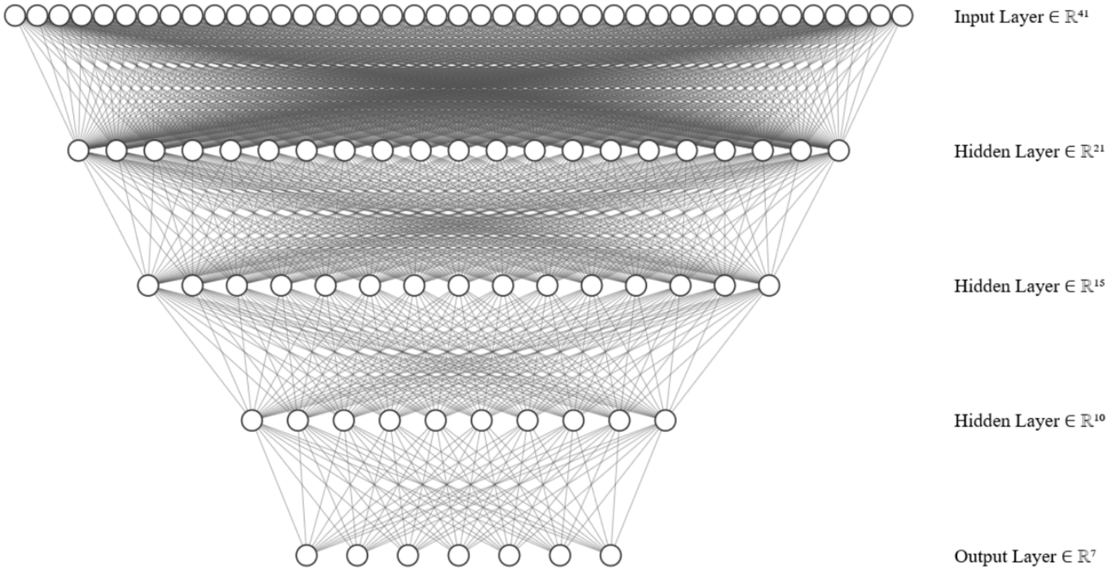

Input Layer $\in \mathbb{R}^{41}$
Hidden Layer $\in \mathbb{R}^{21}$
Hidden Layer $\in \mathbb{R}^{15}$
Hidden Layer $\in \mathbb{R}^{10}$
Output Layer $\in \mathbb{R}^{7}$

Figure 4. ANN architecture.

Before training the ANN, the input feature matrix is normalized, and both the input and the output datasets are divided into a training set, a validation set, and a test set. Approximately 70% of the samples of the dataset are used to train the ANN. The validation set forms 15% of the dataset, and are used to provide an unbiased evaluation of the training data fit while tuning model parameters. The remaining 15% of the dataset is the test set, which is used to evaluate the final model.

The overall architecture of the ANN trained is shown in Figure 4. The network consists of five layers, the input layer with 41 neurons for the input features, three hidden layers with 21, 15, and 10 neurons respectively, and an output layer with 7 neurons for the peak voltage and statistics. All the weights (connections between each neuron) are randomly initialized before the training process starts. The training process starts by feeding a feature vector to the input layer and based on this input the activations of the next layer are calculated. The network calculates the activations of each subsequent layer using the following equation:

$$A^{(1)} = \sigma(WA^{(0)} + b) \tag{2}$$

where $A^{(1)}$ and $A^{(0)}$ are activations of the output layer and the input layer respectively, $W$ is the matrix of weights connecting the input and the subsequent layer, $b$ is the bias vector, and $\sigma$ is the activation function. (3) and (4) are the activation functions used for the hidden layers and the output layer respectively. The linear function (4) at the output layer is better at fitting a function compared to the tan-sigmoid function (3).

After calculating the activations of each neuron, the cost function of the entire network is calculated by comparing the network outputs and actual outputs. This cost function is then used to calculate the gradient using Bayesian regularization backpropagation [11, 12], then the calculated gradient is used to perform gradient descent to update the weights. The network iterates through the entire dataset several times until either expected accuracy is achieved or the performance plateaus.

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{3}$$

$$f(x) = x \tag{4}$$

## 4. RESULTS AND DISCUSSION

The ANN was trained for 500 epochs, and the performance of the network was tested on the test set which included samples that the ANN was not exposed to during the training stage. Figure 5 shows the performance of the network on the training set and the test set in terms of mean squared error (MSE). It can be observed that the network starts with a high MSE as the weights are randomly initialized for the first epoch. Once the network starts training, the MSE drops rapidly. Over the first 100 epochs of training, the MSE drops from 50.29 to 0.1757. The performance of the ANN does not improve substantially after the first 100 epochs, and the drop in the MSE is 0.0238 over the next hundred epochs. After that point, the drop in the MSE plateaus, and the improvement in performance is negligible over the last 300 epochs. The final MSE of 0.1384 over the entire training set shows that the ANN was able to closely fit a function that is able to map the parameters of the input modulated signals, to the peak amplitude and the statistics of the summed signal. The performance plot shows the training set slightly outperforms the test set. This is expected, as the test set consists of samples that the ANN has not seen and is using the trained weights and the trained activations of the neurons to estimate the output.

Figure 6 shows the error histograms with 20 bins for errors after 1 epoch (left) and for errors after 500 epochs (right). The error histogram illustrates how far the outputs of the ANN deviate from the target values. When the ANN is trained for 1 epoch, the histogram is spread out, compared to when the ANN is trained for 500 epochs. There are approximately

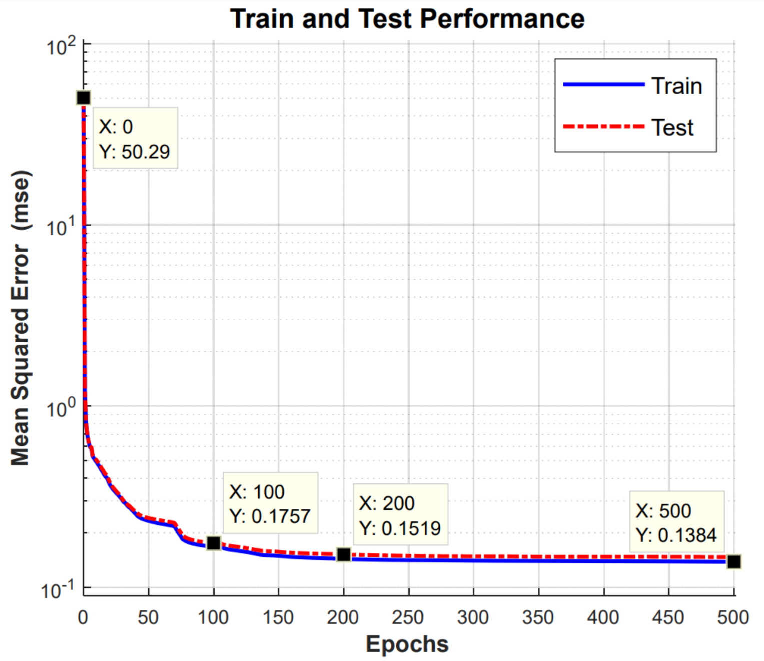


Figure 5. MSE of the Train and Test set plotted against number of epochs.

$10^5$ error instances close to 1 when the neural network is trained only for 1 epoch, but when the network is trained for 500 epochs the number of error instances close to 1 reduces to 20,000.

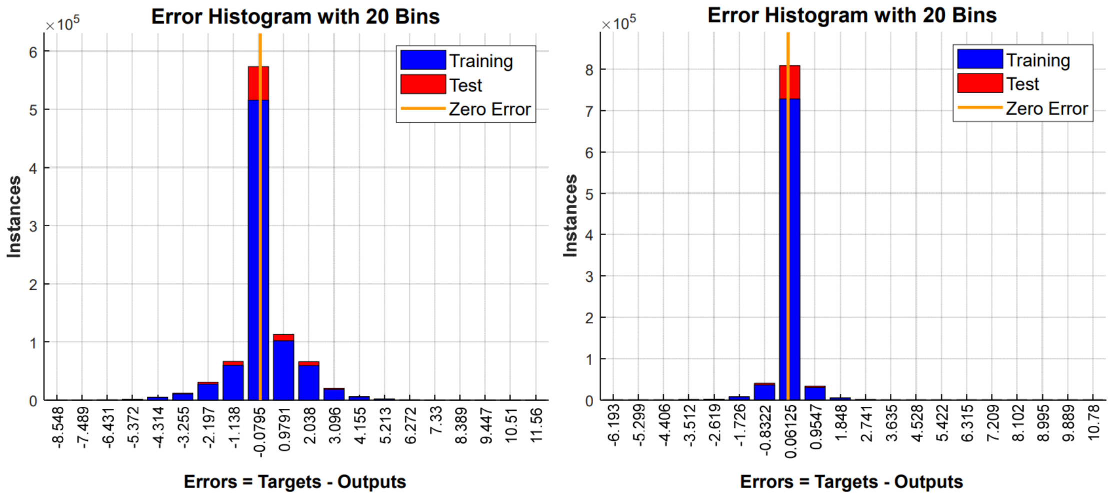


Figure 6. Error histograms after 1 epoch (left) and 500 epochs (right).

The ANN was successfully able to learn a function that maps the parameters of the added signals and the number of signals added to the statistics and the peak amplitude of the summed signal. The trained ANN would be able to estimate the peak amplitude and the statistics of the summed signal based on the parameters of the component signals. The ANN would be able to do this without having to generate the summed signal.

## 5. CONCLUSIONS

This paper demonstrated a way to estimate gain of a summed signal before it goes to a DAC, so that the full range of DAC is used. To accomplish this, a dataset of parameters of the signals added to generate summed signal and corresponding statistics and the peak amplitude of the summed signal was generated. This dataset was used to train an ANN to fit a function that maps the parameters to the statistics and the peak amplitude. This ANN was evaluated on a test set, and was able to closely estimate of the statistics and the peak amplitude. These estimates can be used to decide the amount of gain required for the SDR for optimal performance. Once trained, an ANN would be able to estimate gain based just on the parameters of the signals summed, without having to look at the entire summed signal.

## REFERENCES

[1] Friedrich K Jondral. Software-defined radio—basics and evolution to cognitive radio. *EURASIP journal on wireless communications and networking*, 2005(3):1–9, 2005.

[2] John F Kenney and ES Keeping. *Moments in Standard Units*, volume 1. Van Nostrand Princeton, NJ, USA, 1962.

[3] Stuart Russel, Peter Norvig, et al. *Artificial intelligence: a modern approach*. Pearson Education Limited London, 2013.

[4] Viraj Gajjar and Kurt Kosbar. CSI estimation using artificial neural network. In *International Telemetering Conference*, volume 55, pages 413–422, Las Vegas, NV, USA, 2019.

[5] Vin-sen Feng and Shih Yu Chang. Determination of wireless networks parameters through parallel hierarchical support vector machines. *IEEE Transactions on Parallel and Distributed Systems*, 23(3):505–512, 2012. doi: 10.1109/TPDS.2011.156.

[6] Brad K. Donohoo, Chris Ohlsen, Sudeep Pasricha, Yi Xiang, and Charles Anderson. Context-aware energy enhancements for smart mobile devices. *IEEE Transactions on Mobile Computing*, 13(8):1720–1732, 2014. doi: 10.1109/TMC.2013.94.

[7] Chao-Kai Wen, Shi Jin, Kai-Kit Wong, Jung-Chieh Chen, and Pangan Ting. Channel estimation for massive mimo using gaussian-mixture bayesian learning. *IEEE Transactions on Wireless Communications*, 14(3):1356–1368, 2015. doi: 10.1109/TWC.2014.2365813.

[8] Donald F Specht et al. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[9] Balázs Csanád Csáji et al. Approximation with artificial neural networks. In *MS Thesis*. Faculty of Sciences, Etvs Lornd University, Hungary, 2001.

[10] Bernard Sklar and FJ Harris. *Digital Communications: Fundamentals and Applications*. Prentice-hall, Englewood Cliffs, NJ, 1988.

[11] David JC MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

[12] F. Dan Foresee and M.T. Hagan. Gauss-newton approximation to bayesian learning. In *Proceedings of International Conference on Neural Networks (ICNN'97)*, volume 3, pages 1930–1935 vol.3, 1997. doi: 10.1109/ICNN.1997.614194.

# V. DISTRIBUTION-BASED GAIN ESTIMATION FOR AERONAUTICAL SOFTWARE-DEFINED RADIO APPLICATIONS

Viraj K. Gajjar, and Kurt L. Kosbar
Department of Electrical Engineering
Missouri University of Science and Technology
Rolla, Missouri 65409–0050
Tel: 573–202–1062
Email: vgf4c@mst.edu

## ABSTRACT

An aircraft often uses multiple communication links for aeronautical telecommunications network. This paper investigates a method to estimate the statistical distribution of the time-domain waveform created when multiple baseband signals are summed. This problem arises when multiple communication signals are generated by a single software-defined radio transmitter. The problem is formulated to optimally estimate digital to analog converter's (DAC) gain when shared by multiple modulators, to avoid clipping and to maximize transmitted power. A dataset of typical signals is generated, and multiple statistical distributions are fitted on this summed signal. The generalized gamma distribution (GGD) fits the data particularly well. The parameters of the GGD were then calculated for the entire dataset of summed signals. A deep neural network (DNN) is then trained to estimate the distribution parameters using the component signal's features. The DNN was able to estimate the GGD's parameters without having to generate or observe the summed signal.

**Keywords:** software defined radio, digital to analog converter, gain estimation, neural network

# 1. INTRODUCTION

Future aerial communications requires the following connectivity applications for an airplane: air traffic management, aircraft communication addressing and reporting, single-pilot operations, and massive sensor-data transfer [1]. As shown in Fig. 1, these applications can be facilitated through direct air-to-ground communications, satellite networks, high altitude platform networks, and aircraft to aircraft links [1]. The communication links associated with these applications may require substantial cabling and hardware to generate baseband waveforms. The cabling related costs for an aircraft have been estimated at 2000 USD per kilogram, which leads to a cost ranging from 14 Million USD for an aircraft like an Airbus A320 to 50 Million USD for one like a Boeing B787 [2]. It can be challenging to modify, integrate, troubleshoot and maintain new applications on hardware-based systems and cabling [3]. Using software to generate the baseband waveforms instead of hardware can help address some of these concerns, since it may considerably reduce the amount of hardware and cabling. A software-defined radio (SDR) can be an excellent fit for this application since it can support multiple communication protocols with software versatility. An SDR can either generate or record digital samples and represent them as baseband signals, which can then be up-converted and transmitted using transceiver hardware [4]. Amrhar et. al. demonstrated that an SDR-based integrated modular avionics architecture could significantly reduce the size, weight, power, and cost in an aircraft [5]. SDR platforms on an aircraft would help improve fuel efficiency through reduced hardware and provide flexibility in terms of design and troubleshooting.

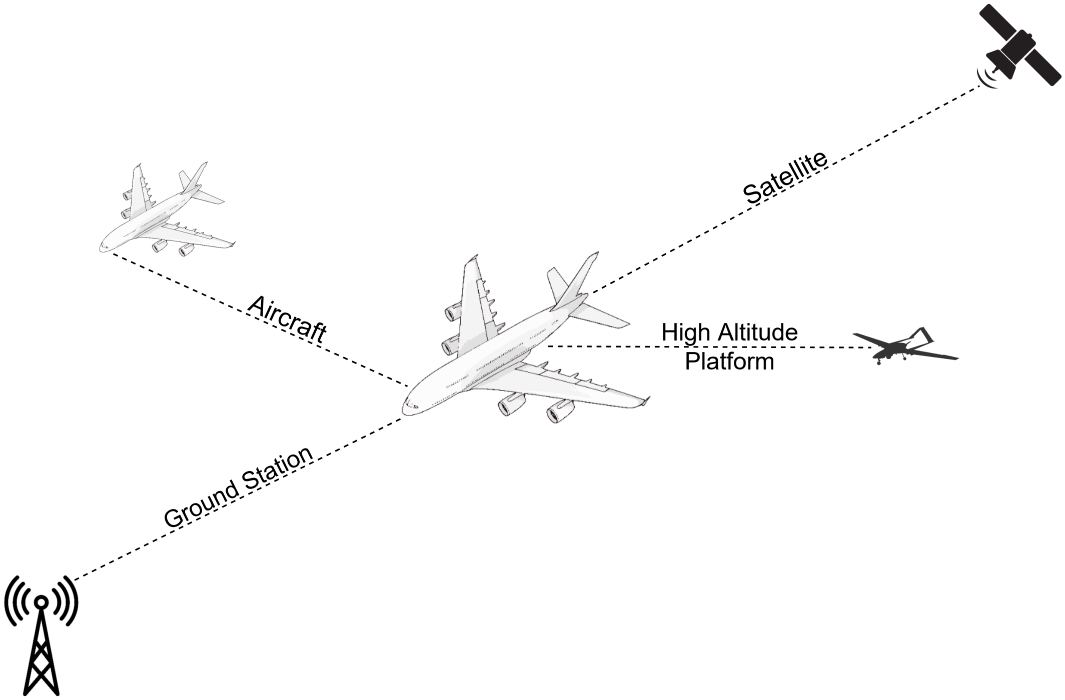


Figure 1. Typical communication links of an aircraft.

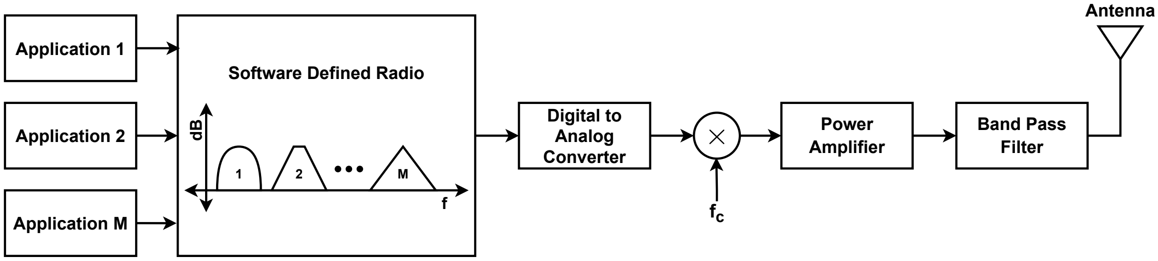


Figure 2. SDR based implementation of an airplane's communication links. Individual hardware for each radio application is replaced by an SDR where all the baseband signals are combined. The combined-digital signal is then passed through a DAC to produce an analog signal, which is then up-converted and transmitted at desired carrier frequency.

Figure 2 demonstrates multiple radio waveform applications implemented using a single SDR. The SDR can generate waveforms for each application then sums them into a single digital waveform. Before up-converting and transmitting the composite waveform, a digital-to-analog converter (DAC) produces an analog waveform. The peak amplitude of the composite waveform would help determine the optimal gain for the DAC that minimizes quantization noise while simultaneously using the full range of the DAC. In some cases, the peak might occur so infrequently that the system might be able to tolerate clipping it, with minimal impact on system performance. A DAC's gain is relatively easy to estimate when dealing with an individual waveform, even after the distortion induced by a pulse-shaping filter. When multiple waveforms are combined, it can become difficult to estimate peak amplitude and the voltage distribution of the composite waveform since each component waveform could potentially have different waveform characteristics such as modulation technique, power level, and bit rate. Some may find it preferable to estimate the statistical distribution of the composite waveform to decide DAC's optimal gain for different use cases.

Deep learning (DL) has grown in popularity in recent years due to dramatic increase in availability of computing power [6, 7]. This success has led to an increased interest in the application of DL in communications and signal processing [8, 9]. For instance DL has been used for automatic modulation classification (AMC) [10, 11]. Peng et al. [10] use signal constellation diagrams as images to train a convolutional neural network (CNN) for AMC. Meng et al. [11] directly use received signals' complex baseband representation to train a CNN for AMC. Ye et al. [12] demonstrate a way to directly recover transmitted symbols in an orthogonal frequency division multiplexing (OFDM) system using DL, without having to explicitly estimate channel state information (CSI). [13] and [14] also show ways to estimate CSI using DL.

A type of DL algorithm, deep neural network (DNN) regression, is a statistical process for estimating relations between input and output variables. For a given D-dimensional vector $\mathbf{x}$ of independent input variables, regression analysis predicts one or more continuous-valued estimation targets [15]. Regression analysis can be used to estimate a mobile terminal's specific usage pattern in diverse spatiotemporal and device contexts [16]. DNN regression also has applications in channel state information estimation [17]. This property of regression analysis to estimate continuous variables can be useful when estimating parameters of a statistical distribution of composite signal's voltage levels.

This article is an extension of [18], where the authors demonstrate a method of estimating gain for a DAC using statistical moments. This article demonstrates that a composite signal generated by summing signals with different modulation techniques, power levels, data rate, and normalized carrier frequency has voltage levels that follow GGD [19]. A dataset of 100 000 composite signals is created, and their respective GGD parameters are calculated. The dataset is then used to train a DNN for regression, where the inputs are modulation technique, power level, data rate, the carrier frequency of each component signal, and the number of signals added. The output targets of the DNN are the parameter values of the respective GGD of the summed signal.

The rest of the article is structured as follows: Section II discusses how the composite signals were generated and how their statistical distribution was determined. Section III demonstrates how DNN regression was used to estimate the summed signal's statistical distribution parameters. Section IV provides simulation results for the proposed multi-output DNN regression technique, and compares these results with other multi-output regression techniques. Finally, Section V provides a conclusion for the paper.

## 2. COMPOSITE SIGNAL'S DISTRIBUTION AND THE DATASET

This section describes how a composite signal is generated using multiple M-ary quadrature amplitude modulated (M-QAM) signals. Then residual sum of squares (RSS) [20], and Kullback-Leibler (KL) divergence [21], are used to determine a theoretical distribution that closely fits the voltage distribution of the summed signal. This theoretical distribution is then used to generate a dataset for training a DNN for estimating the distribution's parameters.

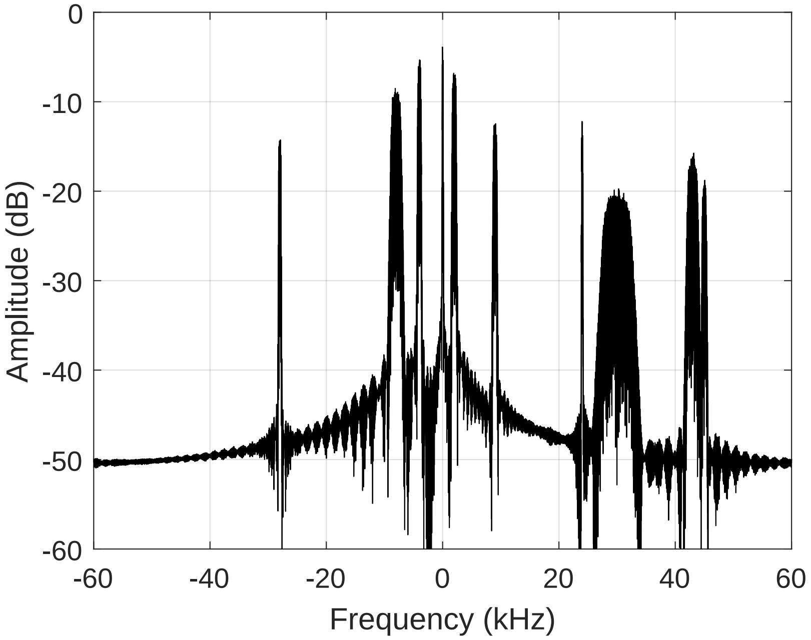


Figure 3. Frequency domain representation of a combined signal generated by adding ten signals. The features of the component signals were randomly selected from Table 1.

## 2.1. THE COMPOSITE SIGNAL

The composite signal is generated by adding multiple component signals with varying modulation orders, data rates, power levels, and normalized carrier frequencies, where number of component signals are chosen randomly between two and ten. The component M-QAM signal has the following form [22]:

$$s_m(t) = A_{m,I}g_T(t)cos2\pi f_ct + A_{m,Q}g_T(t)sin2\pi f_ct, \tag{1}$$

where $s_m(t)$ is the M-QAM signal, $A_{m,I}$ and $A_{m,Q}$ are the in-phase and quadrature amplitude levels, $g_T(t)$ is the pulse shaping filter, $f_c$ is the baseband carrier frequency, and $m = 1, 2, 3, ...M$ is the modulation order of M-QAM signal. Therefore, a composite signal can be represented as follows:

$$c(t) = \sum_{i=1}^{N} p_i s_m(t), \tag{2}$$

where, $c(t)$ is the composite signal, $N$ is the number of M-QAM signals added, and $p_i$ is the power factor.

Table 1. Range of parameters used for generating a combined signal.

| Parmeter | Range |
|---|---|
| Modulation order (M) | 2, 4, 8, 16, 32, 64, 128, and 256 |
| Data rate (bits per second) | 1000 bps to 10000 bps |
| Power level (dBm) | - 30 dBm to 0 dBm |
| Normalized carrier frequency (kHz) | -60 kHz to 60 kHz |

The modulation order, M, of a M-QAM component signal is randomly selected from the eight values listed in Table 1. The data rate, power level, and normalized carrier frequency of the component signal are randomly sampled from the range listed in Table 1. Figure 3 shows the power spectrum of a composite signal generated by adding ten component signals.

## 2.2. DISTRIBUTION FITTING

RSS and KL-divergence were used to determine the statistical distributions that accurately represent the voltage levels of the composite signals. A distribution was fit to the composite signal by choosing optimal parameters that minimize $R_{SS}$ where:

$$R_{SS} = \sum_{k=1}^{n} (y_k - \hat{y_k})^2 \tag{3}$$

where $n$ is the number of data samples, $y_k$ is the $k^{th}$ data sample, and $\hat{y_k}$ is an estimate of $y_k$ from the fitted distribution.

Table 2. RSS and KL-Divergence scores for the four distributions fitted to hundered different combined signals. The scores are expressed as the mean of the scores plus/minus their standard deviation.

| Distribution | $R_{SS}$ <br> $(\mu \pm \sigma)$ | $D_{KL}$ <br> $(\mu \pm \sigma)$ |
|---|---|---|
| Beta | 0.2637±0.3544 | 0.0020±0.0018 |
| Chi | 0.5200±0.6494 | 0.0065±0.0047 |
| Generalized Gamma | 0.0922±0.2189 | 0.0007±0.0009 |
| Rice | 0.2342±0.4192 | 0.0025±0.0031 |

Once the optimal parameters of the fitted distribution were determined, the distribution was compared with the composite signal's samples using the KL-divergence score, also known as relative entropy. For the discrete probability distributions $P$ and $Q$, the KL divergence score from $Q$ to $P$ is defined as:

$$D_{KL}(P \parallel Q) = \sum_{x \in \Phi} P(x) log \left( \frac{P(x)}{Q(x)} \right) \tag{4}$$

where $P$ represents the composite signal's samples, and $Q$ represents theoretical distribution with optimal estimated parameters, and $\Phi$ is the probability space. Table 2 lists four commonly used distributions that reasonably approximate the composite signal, along with the $R_{SS}$ and $D_{KL}$ for each distribution. $R_{SS}$ and $D_{KL}$ were calculated for 1000 composite signals and are represented in terms of their mean $(\mu)$ and the standard deviation $(\sigma)$. Figure 4 shows four distributions with the least $R_{SS}$ when fit to the composite signal with ten signals added.

The distribution with the least $R_{SS}$ and $D_{KL}$ was the generalized gamma distribution (GGD). Therefore, we use GGD to represent the voltage distribution of composite signals. For a non-negative voltage level, $v$, the probability density function (PDF) and the cumulative distribution function (CDF) of $v$ are described in (5) and (6) respectively as

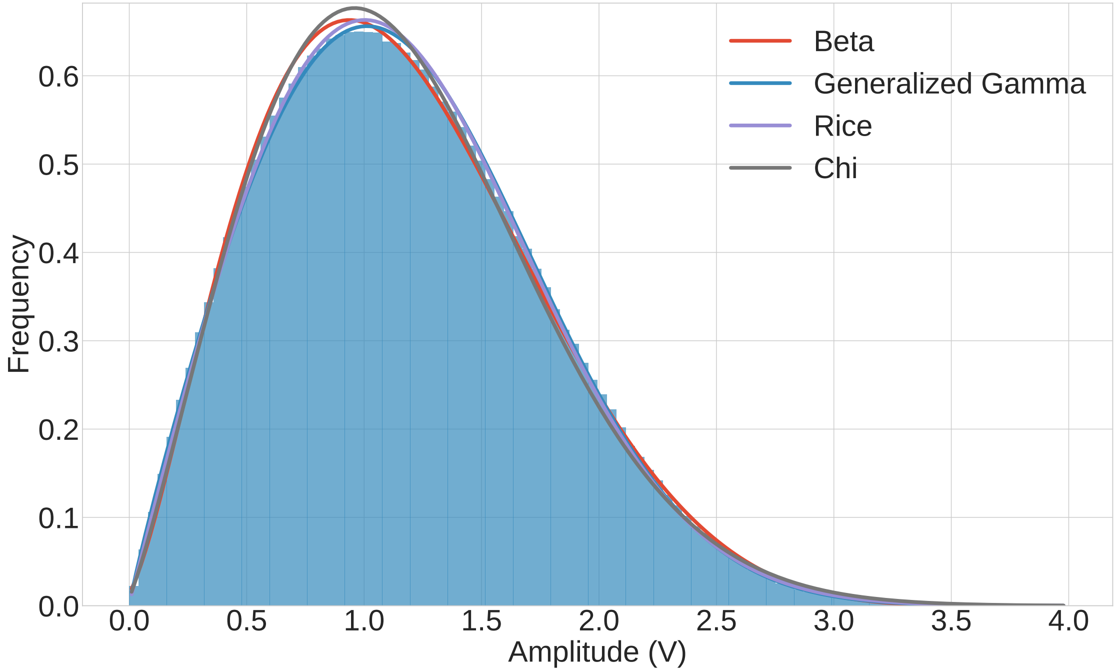


Figure 4. Four common distributions that closely fit the voltage distribution of the combined signal. The combined signal in this plot has ten signals added together.

follows:

$$f(v;s,a,b) = \frac{\left(\frac{b}{s^a}\right) v^{a-1} e^{-\left(\frac{v}{s}\right)^b}}{\Gamma\left(\frac{a}{b}\right)} \tag{5}$$

$$F(v;s,a,b) = \frac{\gamma\left(\frac{a}{b}, \left(\frac{v}{s}\right)^b\right)}{\Gamma\left(\frac{a}{b}\right)} \tag{6}$$

where $s$, $a$, and $b$ are the parameters of the GGD, $\Gamma\left(\frac{a}{b}\right)$ is the gamma function, and $\gamma\left(\frac{a}{b}, \left(\frac{v}{s}\right)^b\right)$ is the lower incomplete gamma function. Estimating the three parameters of the GGD can be used to calculate PDF and CDF the composite signal. This should allow a user to make probability-based decisions for a DAC's gain.

### 2.3. THE DATASET

To train machine learning algorithms for multi-output regression, a dataset $D$ of $N$ samples was simulated, where $N = 10^5$. To simulate $D$, first, $N$ combined signals were generated by summing multiple component signals, where the features of the component signals were selected from Table 1. A minimum of two and a maximum of ten signals were added, and each signal had four features, modulation order, data rate, power level, and normalized carrier frequency. Therefore, each combined signal can be described using $p = 41$ features. 40 features for a maximum of ten signals added, and an extra feature describing the number of signals added. When fewer than ten signals are added the empty features were set to zero.

A GGD distribution was fit on each of the combined signal to calculate GGD's $q = 3$ parameters, shape parameters $a$ and $b$, and a scale parameter $s$. Each sample in $D$ has an input vector of $p$ descriptive variables $\mathbf{x}^{(r)} = (x_1^{(r)}, ..., x_i^{(r)}, ..., x_p^{(r)})$, and an output vector of $q$ target variables $\mathbf{y}^{(r)} = (y_1^{(r)}, ..., y_j^{(r)}, ..., y_q^{(r)})$, where $i \in \{1, ..., p\}$, $j \in \{1, ..., q\}$, and $r \in \{1, ..., N\}$. Overall, the simulated dataset can be described by $N$ samples of of pairs of input and output vectors, i.e., $D = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), ..., (\mathbf{x}^{(N)}, \mathbf{y}^{(N)})\}$.

## 3. DNN REGRESSION

The task is to estimate GGD's parameters by learning a multi-output regression model from $D$. Learning a multi-target regression model consists of finding a function $\Psi$ that assigns $q$ output targets $\mathbf{y}$ for each given input vector $\mathbf{x}$ of length $p$ [23], as described in (7).

$$\begin{aligned}\Psi : S_{X_1} \times \,...\, \times S_{X_p} &\longrightarrow S_{Y_1} \times \,...\, \times S_{Y_q} \\ \mathbf{x} = (x_1, ..., x_p) &\mapsto \mathbf{y} = (y_1, ..., y_q),\end{aligned} \tag{7}$$

where $S_{X_i}$ denotes the sample space of $X_i$, $\forall i \in \{1, ..., p\}$ and $S_{Y_j}$ denotes the sample space of $Y_j$, $\forall j \in \{1, ..., q\}$.

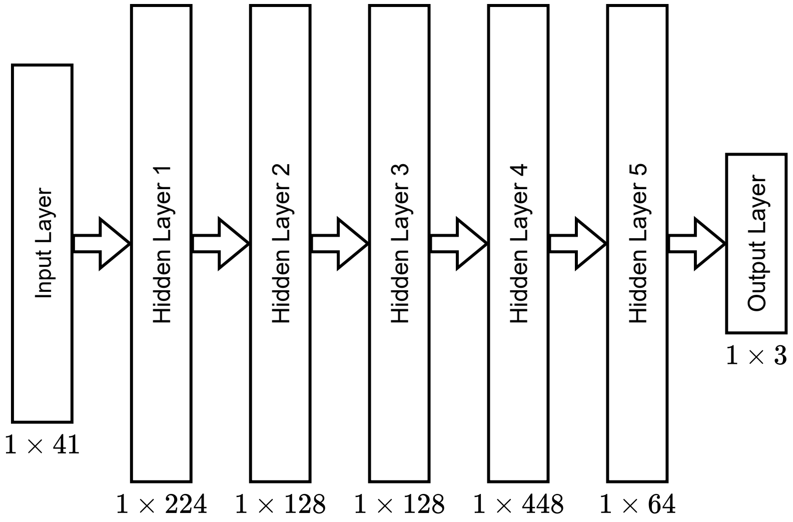


Figure 5. DNN used to estimate voltage distribution of the combined signal. Features of the combined signal are fed to the input layer and the output layer measures the parameters of the GGD.

A DNN was used as a multi-output regression technique for GGD's parameter estimation. As shown in Fig. 5, the DNN has an input dimension of $p = 41$ for input feature vector $\mathbf{x}$. The input layer is followed by five dense/hidden layers with 224, 128, 128, 448, and 64 units respectively. The number of units used in these layers were determined by using a hyperparameteric search. The final dense layer is followed by an output layer with $q = 3$ neurons for estimating target vector $\mathbf{y}$, the estimates are denoted as $\hat{\mathbf{y}}$.

All the dense layers, the input layer, and the output layer were preceded by a batch-normalization layer to standardize inputs to a layer for each mini-batch. Mini-batch size was set to 1000, and an exponentially decaying learning rate was used with an initial learning rate of 0.01. The drop-out rate of the neurons in the network was set to 0.5 to help mitigate overfitting to the training data. The dense layers had ReLU activation function that thresholds the raw scores $o_i$ by 0, i.e. $f(o) = max(0, o_i)$, and the output layer had a linear activation function that does not alter the raw scores, i.e. $f(o) = o_i$.

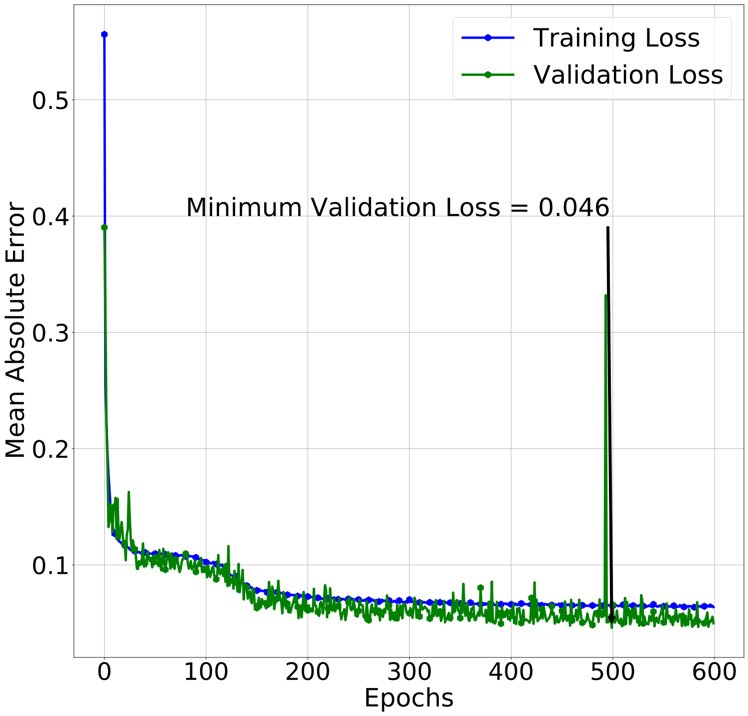


Figure 6. MAE training and validation loss trends. Minimum validation loss is reached on $350^{th}$ epoch.

The dataset was split into three parts to train and test the DNN, 80% for training, 10% for validation, and 10% for testing. Mean Absolute Error (MAE) was used to find errors between the target values and the predictions at the output layer. The error values were then used to perform gradient descent to update the network's weights. Early stopping with the patience of 100 and precision of $10^{-4}$ was used to stop the training based on the validation set's MAE, as shown in Fig. 6.

We noticed that the MAE decreases rapidly in the first few epochs, followed by a more gradual descent until 200 epochs had elapsed. After 200 epochs, the loss plateaus and a minimum validation MAE of 0.0478 on the validation set was reached on the $350^{th}$ epoch. After the training was completed, the model weights corresponding to minimum validation MAE were used for testing the DNN performance on the test set.

# 4. RESULTS AND ANALYSIS

## 4.1. PERFORMANCE OF THE DNN

The following metrics were used to assess GGD's parameters' estimates obtained using a DNN: $R^2$ value, explained variance, mean-squared error (MSE), and MAE. Table 3 summarizes the metric scores for each distribution parameter, along with the overall scores on all three parameters. The DNN is able to fit the shape parameter $s$ the best, with $R^2 = 0.9941$ and $MAE = 0.0241$ obtained on the test set. It can also be observed that the $R^2$ and explained variances scores are similar, indicating that the mean of the error is close to zero. This suggests the estimation of the distribution parameters by the DNN is largely unbiased, which implies that the DNN is less prone to overestimation or underestimation. Figure 7 shows the $R^2$ plot for each of the estimated parameters. It can be observed that the DNN has a difficult time estimating values of parameter $a$ which are larger than 0.8. Also, the $R^2$ plots of shape parameters $a$ and $b$ are more scattered than the scale parameter $s$. Parameter $b$ contributes the most to the overall MAE and MSE, which could be caused by the larger target values of the parameter that leads to larger error values.

Table 3. $R^2$, explained variance, MAE, and MSE performance scores obtained by the proposed DNN for each parameter, and for all three parameters combined.

| Parameter | $R^2$ | Explained Variance | MAE | MSE |
|---|---|---|---|---|
| a | 0.8546 | 0.8547 | 0.0256 | 0.0019 |
| b | 0.9338 | 0.9348 | 0.0919 | 0.0275 |
| s | 0.9941 | 0.9943 | 0.0241 | 0.0011 |
| **All parameters** | **0.9275** | **0.9279** | **0.0472** | **0.0102** |

Figure 8 demonstrates how the amount of data used for training affects the accuracy of the DNN on the hold-out test set. The MAE rapidly improves as the amount of data used for training increases. The improvement in the MAE becomes less significant as more than

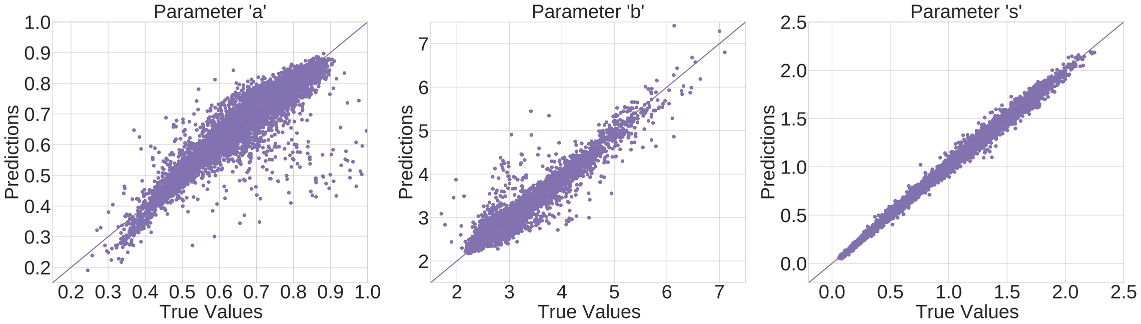


Figure 7. $R^2$ plot for each of the individual parameters of the GGD, estimated using the DNN. $a$ and $b$ are the shape parameters of the distribution, and $s$ is the scale parameter.

80% of the data is used for training. However, this is a minor improvement as the remaining 20% of the data is used for training. This implies that the system's accuracy can potentially be improved with more training data.

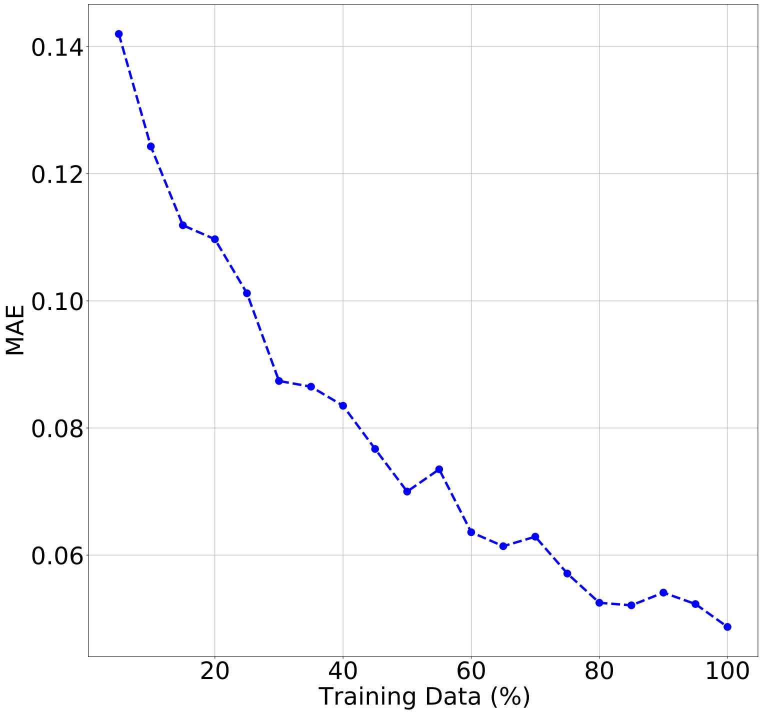


Figure 8. Trend in MAE on the hold-out test set as the amount of data used for training in the DNN is altered.

## 4.2. COMPARISON WITH OTHER TECHNIQUES

Table 4 shows the comparison of $R^2$ scores of five other multi-output regression techniques with the DNN model proposed in this paper. The five techniques, along with the one used in this paper, are listed as follows:

1. Linear Regression (LR) [24]
2. K-Neighbors Regression (K-NR) [25]
3. Support Vector Regression (SVR) [26]
4. Decision Tree Regression (DTR) [27]
5. Shallow Neural Network (SNN) Regression
6. DNN Regression

Table 4. Comparison of $R^2$ obtained using different multiple-regression techniques.

| Parameter | Linear Regression | K-Neighbors Regression | Support Vector Regression | Decision Tree Regression | SNN Regression | DNN Regression |
|---|---|---|---|---|---|---|
| a | 0.4608 | 0.4672 | 0.5406 | 0.4783 | 0.6023 | 0.8546 |
| b | 0.3928 | 0.4734 | 0.5027 | 0.6676 | 0.6719 | 0.9338 |
| s | 0.6817 | 0.6310 | 0.9379 | 0.8773 | 0.9843 | 0.9941 |
| **All parameters** | **0.5118** | **0.5239** | **0.6604** | **0.6744** | **0.7528** | **0.9275** |

The DNN proposed in this paper compares favorably with the competing methods. Considering the first four techniques, LR has the least overall $R^2$ score but has a comparable $R^2$ for parameter $a$ and a better $R^2$ for parameter $s$ compared to K-NR. Overall, both neural networks outperform other multi-output regression techniques.

The DNN implemented with 998 neurons, and five hidden layers would be computationally expensive to train, so its performance was compared with SNN. The SNN only had 100 neurons and three hidden layers, 50 neurons in the first, 30 neurons in the second, and 20 neurons in the third hidden layer. The SNN obtains an overall $R^2 = 0.7528$ by using only one-tenth of the number of neurons used in the DNN, which is 18.83% less than $R^2$ obtained using the DNN. The performance of the SNN is significantly lower when estimating shape parameters of the distribution, $a$, and $b$. All the regression techniques have a harder time estimating the shape parameters of the distribution when compared to the scale parameter $s$. Therefore, a deeper network is required to estimate the shape of a distribution accurately.

### 4.3. A TYPICAL EXAMPLE

To demonstrate an example of a normative scenario, consider a composite signal generated by adding the signals described in Table 5. Figure 9 shows the cumulative distribution function (CDF) of the composite signal's voltage levels, estimated using the DNN. Three different scenarios are listed in Fig. 9 about the probability of exceeding a certain voltage level. For instance, 1% of the voltage levels in the composite signal are above 2.29 volts. This information can be used while estimation the gain of a DAC. For example, if the range of a DAC is $\alpha$ volts and $G$ is the gain, then for $G = \frac{\alpha}{2.29}$ the DAC will clip 1% of the signal samples.

Table 5. Features of four component signals used to generate a combined signal.

| Modulation Order | Data rate (bps) | Power level (dBm) | Carrier frequency (kHz) |
|---|---|---|---|
| 4 | 1544 | -7.59 | 17 |
| 128 | 3906 | -0.05 | 23 |
| 32 | 1707 | -17.39 | 31 |
| 4 | 7670 | -9.25 | -33 |
| 256 | 2846 | -11.29 | 34 |
| 128 | 2639 | -23.92 | 6 |

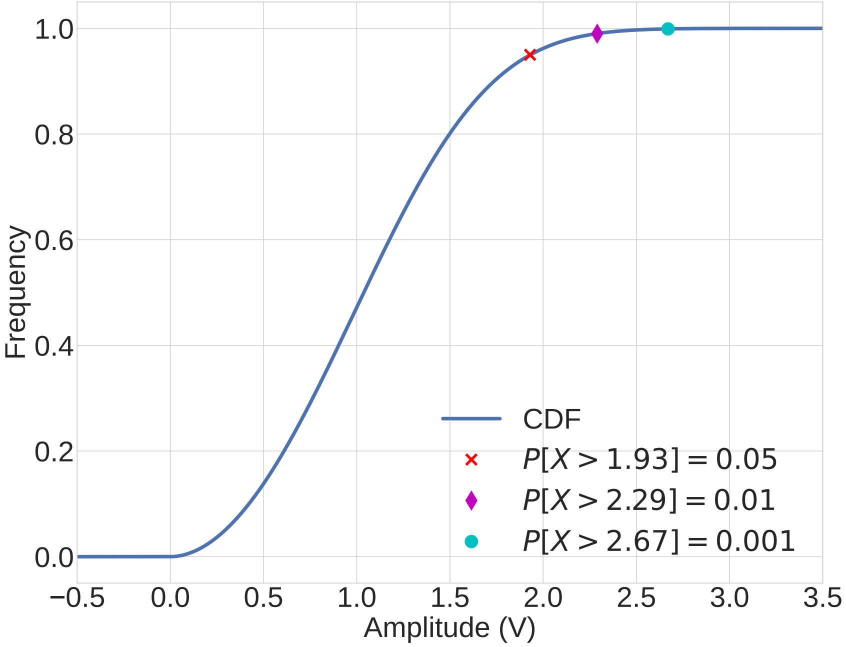


Figure 9. CDF of the combined signal estimated using the DNN. The features of the component signals summed to form the combined signal are listed in Table 5.

## 5. CONCLUSION

Aiming to estimate the optimal gain of a DAC when an SDR is used to implement multiple baseband signals in an aircraft, a deep learning based approach to estimate the voltage distribution of the summed signal to be transmitted is proposed. An analytical distribution that accurately describes the voltage distribution of the composite signal is found based on the RSS fitting method. It appears that the composite signal closely follows a GGD. A DNN was trained to estimate GGD's parameters using simulated data of summed signals. Once a DNN is trained, the estimated distribution should allow a user to estimate DAC's gain based on the requirement.

# REFERENCES

[1] Aygün Baltaci, Ergin Dinc, Mustafa Ozger, Abdulrahman Alabbasi, Cicek Cavdar, and Dominic Schupke. A survey of wireless networks for future aerial communications (facom). *IEEE Communications Surveys Tutorials*, 23(4):2833–2884, 2021. doi: 10.1109/COMST.2021.3103044.

[2] C. Furse and R. Haupt. Down to the wire [aircraft wiring]. *IEEE Spectrum*, 38(2): 34–39, 2001. doi: 10.1109/6.898797.

[3] Ramiro Sámano-Robles, Eduardo Tovar, João Cintra, and André Rocha. Wireless avionics intra-communications: Current trends and design issues. In *2016 Eleventh International Conference on Digital Information Management (ICDIM)*, pages 266–273. IEEE, 2016.

[4] Nathan Price and Kurt Kosbar. Decoupling hardware and software concerns in aircraft telemetry sdr systems. In *International Telemetering Conference*, volume 54, pages 1–11, 2018.

[5] Abdessamad Amrhar, Alireza Avakh Kisomi, Eric Zhang, Joe Zambrano, Claude Thibeault, and René Landry. Multi-mode reconfigurable software defined radio architecture for avionic radios. In *2017 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, pages 2D1–1. IEEE, 2017.

[6] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

[7] Shi Dong, Ping Wang, and Khushnood Abbas. A survey on deep learning and its applications. *Computer Science Review*, 40:100379, 2021.

[8] Reinaldo Padilha França, Ana Carolina Borges Monteiro, Rangel Arthur, and Yuzo Iano. An overview of deep learning in big data, image, and signal processing in the modern digital age. *Trends in Deep Learning Methodologies*, pages 63–87, 2021.

[9] Linglong Dai, Ruicheng Jiao, Fumiyuki Adachi, H. Vincent Poor, and Lajos Hanzo. Deep learning for wireless communications: An emerging interdisciplinary paradigm. *IEEE Wireless Communications*, 27(4):133–139, 2020. doi: 10.1109/MWC.001.1900491.

[10] Shengliang Peng, Hanyu Jiang, Huaxia Wang, Hathal Alwageed, Yu Zhou, Marjan Mazrouei Sebdani, and Yu-Dong Yao. Modulation classification based on signal constellation diagrams and deep learning. *IEEE transactions on neural networks and learning systems*, 30(3):718–727, 2018.

[11] Fan Meng, Peng Chen, Lenan Wu, and Xianbin Wang. Automatic modulation classification: A deep learning enabled approach. *IEEE Transactions on Vehicular Technology*, 67(11):10760–10772, 2018.

[12] Hao Ye, Geoffrey Ye Li, and Biing-Hwang Juang. Power of deep learning for channel estimation and signal detection in ofdm systems. *IEEE Wireless Communications Letters*, 7(1):114–117, 2017.

[13] Mehran Soltani, Vahid Pourahmadi, Ali Mirzaei, and Hamid Sheikhzadeh. Deep learning-based channel estimation. *IEEE Communications Letters*, 23(4):652–655, 2019.

[14] Changqing Luo, Jinlong Ji, Qianlong Wang, Xuhui Chen, and Pan Li. Channel state information prediction for 5G wireless communications: A deep learning approach. *IEEE Transactions on Network Science and Engineering*, 7(1):227–236, 2020. doi: 10.1109/TNSE.2018.2848960.

[15] Chunxiao Jiang, Haijun Zhang, Yong Ren, Zhu Han, Kwang-Cheng Chen, and Lajos Hanzo. Machine learning paradigms for next-generation wireless networks. *IEEE Wireless Communications*, 24(2):98–105, 2016.

[16] Brad K Donohoo, Chris Ohlsen, Sudeep Pasricha, Yi Xiang, and Charles Anderson. Context-aware energy enhancements for smart mobile devices. *IEEE Transactions on Mobile Computing*, 13(8):1720–1732, 2013.

[17] Viraj Gajjar and Kurt Kosbar. CSI estimation using artificial neural network. In *International Telemetering Conference*, volume 55, pages 413–422, Las Vegas, NV, USA, 2019.

[18] Viraj Gajjar and Kurt Kosbar. Rapid gain estimation for multi-user software defined radio applications. In *International Telemetering Conference*, volume 56, Las Vegas, NV, USA, 2020.

[19] Edney W Stacy. A generalization of the gamma distribution. *The Annals of mathematical statistics*, 33(3):1187–1192, 1962.

[20] Thomas J Archdeacon. *Correlation and regression analysis: A historian's guide*. Univ of Wisconsin Press, 1994.

[21] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[22] John G Proakis, Masoud Salehi, Ning Zhou, and Xiaofeng Li. *Communication systems engineering*, volume 2. Prentice Hall New Jersey, 1994.

[23] Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larranaga. A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5):216–233, 2015.

[24] Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[25] Oliver Kramer. *Dimensionality reduction with unsupervised nearest neighbors*, volume 51. Springer, 2013.

[26] M. Brudnak. Vector-valued support vector regression. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 1562–1569, 2006. doi: 10.1109/IJCNN.2006.246619.

[27] Glenn De'Ath. Multivariate regression trees: a new technique for modeling species–environment relationships. *Ecology*, 83(4):1105–1117, 2002.

# SECTION

## 2. CONCLUSIONS AND FUTURE DIRECTIONS

In this dissertation, applications of ML in the fields of computer vision and digital communication are presented. Plant identification systems are developed using two approaches: HC features extracted from leaf images to train an SVM classifier and end-to-end CNNs for leaf classification. An ANN-based system was developed to estimate CSI for a MIMO by using atmospheric conditions as features. Finally, ANN-based gain estimation techniques were presented for aeronautical multi-user SDR applications.

### 2.1. CONCLUSIONS

The first two papers developed ML-based plant identification systems by classifying leaf images. Simulation results of the first paper show that the proposed HC feature extraction technique can achieve state-of-the-art accuracies of 97.91% and 99.56% on two publicly available leaf datasets, Flavia and Swedish, respectively. The second paper proposed the F2LSM leaf dataset formed by combining five publicly available leaf datasets. The F2LSM dataset is available for download at `https://scholarsmine.mst.edu/research_data/8/`. The best performing CNN model in the second paper, Efficient B6, achieved a classification accuracy of 95.93% before oversampling and undersampling were performed to mitigate class imbalance and classification accuracy of 98.41% after oversampling and undersampling.

The third paper's simulation results show that an ANN can successfully learn a function that can map the parameters affecting the CSI of a MIMO communication link to CSI coefficients. Although the ANN was only tested on the simulated dataset, sufficient data and a deeper network can make this technique viable for real-world applications.

The fourth and fifth papers proposed DL-based approaches to estimate the voltage distribution of a summed signal generated by an SDR for multiple aircraft applications. The fourth paper's simulation results suggest that an ANN can learn a function that maps the features of the component signals in the combined signal to the statistics and the peak of the combined signal. In the fifth paper, it was demonstrated that the combined signal follows GGD. Simulation results showed that ANN was able to estimate GGD's shape and scale parameters with an overall $R^2 = 0.93$ and that ANN outperforms other multi-output regression techniques for this particular case.

## 2.2. FUTURE DIRECTIONS

The F2LSM dataset proposed for plant identification can be further expanded by including more species. The CNNs trained for classification in the F2LSM dataset were tested on test images sampled from the F2LSM dataset itself. Future work may include training ML algorithms on the F2LSM dataset and then testing them on a test set not sampled from F2LSM dataset. For CSI and gain estimation, it should be noted that the ANN models were trained on simulated data generated under static conditions. In the future, the effectiveness of the techniques proposed should be dynamically tested in a real-world environment.

# REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[2] Abdessamad Amrhar, Alireza Avakh Kisomi, Eric Zhang, Joe Zambrano, Claude Thibeault, and René Landry. Multi-mode reconfigurable software defined radio architecture for avionic radios. In *2017 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, pages 2D1–1. IEEE, 2017.

[3] Voncarlos M. Araújo, Alceu S. Britto Jr., Luiz S. Oliveira, and Alessandro L. Koerich. Two-view fine-grained classification of plant species. *Neurocomputing*, 467:427–441, 2022. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2021.10.015. URL https://www.sciencedirect.com/science/article/pii/S0925231221014934.

[4] Thomas J Archdeacon. *Correlation and regression analysis: A historian's guide*. Univ of Wisconsin Press, 1994.

[5] O.H. Babatunde, L. Armstrong, J. Leng, and D. Diepeveen. A survey of computer-based vision systems for automatic identification of plant species. *Journal of Agricultural Informatics*, 6(1):61–71, 2015. doi: 10.17700/jai.2015.6.1.152. URL https://researchrepository.murdoch.edu.au/id/eprint/25987/.

[6] Qinbo Bai, Jintao Wang, Yue Zhang, and Jian Song. Deep learning-based channel estimation algorithm over time selective fading channels. *IEEE Transactions on Cognitive Communications and Networking*, 6(1):125–134, 2020. doi: 10.1109/TCCN.2019.2943455.

[7] Aygün Baltaci, Ergin Dinc, Mustafa Ozger, Abdulrahman Alabbasi, Cicek Cavdar, and Dominic Schupke. A survey of wireless networks for future aerial communications (facom). *IEEE Communications Surveys Tutorials*, 23(4):2833–2884, 2021. doi: 10.1109/COMST.2021.3103044.

[8] Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.*, 6(1):20–29, jun 2004. ISSN 1931-0145. doi: 10.1145/1007730.1007735. URL https://doi.org/10.1145/1007730.1007735.

[9] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[10] Bradley C Bennett and Michael J Balick. Phytomedicine 101: plant taxonomy for preclinical and clinical medicinal plant researchers. *Journal of the Society for Integrative Oncology*, 6(4):150, 2008.

[11] Bradley C. Bennett and Michael J. Balick. Does the name really matter? the importance of botanical nomenclature and plant taxonomy in biomedical research. *Journal of Ethnopharmacology*, 152(3):387–392, 2014. ISSN 0378-8741. doi: https://doi.org/10.1016/j.jep.2013.11.042. URL `https://www.sciencedirect.com/science/article/pii/S0378874113008465`.

[12] Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larranaga. A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5):216–233, 2015.

[13] M. Brudnak. Vector-valued support vector regression. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 1562–1569, 2006. doi: 10.1109/IJCNN.2006.246619.

[14] Rainer W. Bussmann. Chapter 4 - Taxonomy–An irreplaceable tool for validation of herbal medicine. In Pulok K. Mukherjee, editor, *Evidence-Based Validation of Herbal Medicine*, pages 87–118. Elsevier, Boston, 2015. ISBN 978-0-12-800874-4. doi: https://doi.org/10.1016/B978-0-12-800874-4.00004-0. URL `https://www.sciencedirect.com/science/article/pii/B9780128008744000040`.

[15] Ayan Chaudhury and John L. Barron. Plant species identification from occluded leaf images. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17 (3):1042–1055, 2020. doi: 10.1109/TCBB.2018.2873611.

[16] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[17] Francois Chollet et al. Keras, 2015. URL `https://github.com/fchollet/keras`.

[18] Chang-Jae Chun, Jae-Mo Kang, and Il-Min Kim. Deep learning-based channel estimation for massive mimo systems. *IEEE Wireless Communications Letters*, 8(4): 1228–1231, 2019. doi: 10.1109/LWC.2019.2912378.

[19] Balázs Csanád Csáji et al. Approximation with artificial neural networks. In *MS Thesis*. Faculty of Sciences, Etvs Lornd University, Hungary, 2001.

[20] Balázs Csanád Csáji et al. Approximation with artificial neural networks. In *MS Thesis*. Faculty of Sciences, Etvs Lornd University, Hungary, 2001.

[21] Linglong Dai, Ruicheng Jiao, Fumiyuki Adachi, H. Vincent Poor, and Lajos Hanzo. Deep learning for wireless communications: An emerging interdisciplinary paradigm. *IEEE Wireless Communications*, 27(4):133–139, 2020. doi: 10.1109/MWC.001.1900491.

[22] F. Dan Foresee and M.T. Hagan. Gauss-newton approximation to bayesian learning. In *Proceedings of International Conference on Neural Networks (ICNN'97)*, volume 3, pages 1930–1935 vol.3, 1997. doi: 10.1109/ICNN.1997.614194.

[23] Glenn De'Ath. Multivariate regression trees: a new technique for modeling species–environment relationships. *Ecology*, 83(4):1105–1117, 2002.

[24] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[25] Shi Dong, Ping Wang, and Khushnood Abbas. A survey on deep learning and its applications. *Computer Science Review*, 40:100379, 2021.

[26] Brad K Donohoo, Chris Ohlsen, Sudeep Pasricha, Yi Xiang, and Charles Anderson. Context-aware energy enhancements for smart mobile devices. *IEEE Transactions on Mobile Computing*, 13(8):1720–1732, 2013.

[27] Brad K. Donohoo, Chris Ohlsen, Sudeep Pasricha, Yi Xiang, and Charles Anderson. Context-aware energy enhancements for smart mobile devices. *IEEE Transactions on Mobile Computing*, 13(8):1720–1732, 2014. doi: 10.1109/TMC.2013.94.

[28] Brad K. Donohoo, Chris Ohlsen, Sudeep Pasricha, Yi Xiang, and Charles Anderson. Context-aware energy enhancements for smart mobile devices. *IEEE Transactions on Mobile Computing*, 13(8):1720–1732, 2014. doi: 10.1109/TMC.2013.94.

[29] D S Fabricant and N R Farnsworth. The value of plants used in traditional medicine for drug discovery. *Environmental Health Perspectives*, 109(suppl 1):69–75, 2001. doi: 10.1289/ehp.01109s169.

[30] Vin-sen Feng and Shih Yu Chang. Determination of wireless networks parameters through parallel hierarchical support vector machines. *IEEE Transactions on Parallel and Distributed Systems*, 23(3):505–512, 2012. doi: 10.1109/TPDS.2011.156.

[31] Vin-sen Feng and Shih Yu Chang. Determination of wireless networks parameters through parallel hierarchical support vector machines. *IEEE Transactions on Parallel and Distributed Systems*, 23(3):505–512, 2012. doi: 10.1109/TPDS.2011.156.

[32] Alberto Fernández, Salvador Garcia, Francisco Herrera, and Nitesh V Chawla. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, 61:863–905, 2018.

[33] Reinaldo Padilha França, Ana Carolina Borges Monteiro, Rangel Arthur, and Yuzo Iano. An overview of deep learning in big data, image, and signal processing in the modern digital age. *Trends in Deep Learning Methodologies*, pages 63–87, 2021.

[34] C. Furse and R. Haupt. Down to the wire [aircraft wiring]. *IEEE Spectrum*, 38(2): 34–39, 2001. doi: 10.1109/6.898797.

[35] Viraj Gajjar and Kurt Kosbar. CSI estimation using artificial neural network. In *International Telemetering Conference*, volume 55, pages 413–422, Las Vegas, NV, USA, 2019.

[36] Viraj Gajjar and Kurt Kosbar. CSI estimation using artificial neural network. In *International Telemetering Conference*, volume 55, pages 413–422, Las Vegas, NV, USA, 2019.

[37] Viraj Gajjar and Kurt Kosbar. Rapid gain estimation for multi-user software defined radio applications. In *International Telemetering Conference*, volume 56, Las Vegas, NV, USA, 2020.

[38] Viraj Gajjar, Ze-Hao Lai, and Kurt Kosbar. Fast classification of leaf images for agricultural remote sensing applications. In *International Telemetering Conference*, volume 54, pages 569–578, 2018.

[39] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. " O'Reilly Media, Inc.", 2019.

[40] Hengtao He, Chao-Kai Wen, Shi Jin, and Geoffrey Ye Li. Deep learning-based channel estimation for beamspace mmwave massive mimo systems. *IEEE Wireless Communications Letters*, 7(5):852–855, 2018.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[42] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[43] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[44] Jiang Huixian. The analysis of plants image recognition based on deep learning and artificial neural network. *IEEE Access*, 8:68828–68841, 2020. doi: 10.1109/ACCESS.2020.2986946.

[45] Cholhong Im, Hirobumi Nishida, and Tosiyasu L Kunii. A hierarchical method of recognizing plant species by leaf shapes. In *Proceedings of IAPR Workshop on Machine Vision Applications (MVA 1998)*, pages 158–161. Citeseer, 1998.

[46] ITU Radiocommunication Sector. P.838 : Specific attenuation model for rain for use in prediction methods. `https://www.itu.int/rec/R-REC-P.838/en`, 2005.

[47] ITU Radiocommunication Sector. P.676 : Attenuation by atmospheric gases. `https://www.itu.int/rec/R-REC-P.676`, 2013.

[48] ITU Radiocommunication Sector. P.840 : Attenuation due to clouds and fog. `https://www.itu.int/rec/R-REC-P.840`, 2013.

[49] Chunxiao Jiang, Haijun Zhang, Yong Ren, Zhu Han, Kwang-Cheng Chen, and Lajos Hanzo. Machine learning paradigms for next-generation wireless networks. *IEEE Wireless Communications*, 24(2):98–105, 2016.

[50] Chunxiao Jiang, Haijun Zhang, Yong Ren, Zhu Han, Kwang-Cheng Chen, and Lajos Hanzo. Machine learning paradigms for next-generation wireless networks. *IEEE Wireless Communications*, 24(2):98–105, 2016.

[51] Friedrich K Jondral. Software-defined radio—basics and evolution to cognitive radio. *EURASIP journal on wireless communications and networking*, 2005(3):1–9, 2005.

[52] Abdul Kadir, Lukito Nugroho, Adhi Susanto, and Paulus Santosa. Experiments of zernike moments for leaf identification. *Journal of Theoretical and Applied Information Technology*, 41:82–93, 07 2012.

[53] Abdul Kadir, Lukito Edi Nugroho, Adhi Susanto, and Paulus Insap Santosa. Performance improvement of leaf identification system using principal component analysis. *International Journal of Advanced Science and Technology*, 44(11):113–124, 2012.

[54] John F Kenney and ES Keeping. *Moments in Standard Units*, volume 1. Van Nostrand Princeton, NJ, USA, 1962.

[55] Oliver Kramer. *Dimensionality reduction with unsupervised nearest neighbors*, volume 51. Springer, 2013.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL `https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`.

[57] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[58] Munish Kumar, Surbhi Gupta, Xiao-Zhi Gao, and Amitoj Singh. Plant species recognition using morphological features and adaptive boosting methodology. *IEEE Access*, 7:163912–163918, 2019. doi: 10.1109/ACCESS.2019.2952176.

[59] Neeraj Kumar, Peter N. Belhumeur, Arijit Biswas, David W. Jacobs, W. John Kress, Ida C. Lopez, and João V. B. Soares. Leafsnap: A computer vision system for automatic plant species identification. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 502–516, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-33709-3.

[60] Neeraj Kumar, Peter N. Belhumeur, Arijit Biswas, David W. Jacobs, W. John Kress, Ida C. Lopez, and João V. B. Soares. Leafsnap: A computer vision system for automatic plant species identification. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 502–516, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-33709-3.

[61] Ping Lang, Xiongjun Fu, Marco Martorella, Jian Dong, Rui Qin, Xianpeng Meng, and Min Xie. A comprehensive survey of machine learning applied to radar signal processing. 2020. doi: 10.48550/ARXIV.2009.13702. URL `https://arxiv.org/abs/2009.13702`.

[62] Sue Han Lee, Chee Seng Chan, Paul Wilkin, and Paolo Remagnino. Deep-plant: Plant identification with convolutional neural networks. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 452–456, 2015. doi: 10.1109/ICIP.2015.7350839.

[63] G. Lera and M. Pinzolas. Neighborhood based levenberg-marquardt algorithm for neural network training. *IEEE Transactions on Neural Networks*, 13(5):1200–1203, 2002. doi: 10.1109/TNN.2002.1031951.

[64] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *2011 International Conference on Computer Vision*, pages 2548–2555, 2011. doi: 10.1109/ICCV.2011.6126542.

[65] Zhenbo Li, Ruohao Guo, Meng Li, Yaru Chen, and Guangyao Li. A review of computer vision technologies for plant phenotyping. *Computers and Electronics in Agriculture*, 176:105672, 2020. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2020.105672. URL `https://www.sciencedirect.com/science/article/pii/S0168169920307511`.

[66] Konstantinos G. Liakos, Patrizia Busato, Dimitrios Moshou, Simon Pearson, and Dionysis Bochtis. Machine learning in agriculture: A review. *Sensors*, 18(8), 2018. ISSN 1424-8220. doi: 10.3390/s18082674. URL `https://www.mdpi.com/1424-8220/18/8/2674`.

[67] Wenyi Lin, Kyle Hasenstab, Guilherme Moura Cunha, and Armin Schwartzman. Comparison of handcrafted features and convolutional neural networks for liver mr image adequacy assessment. *Scientific Reports*, 10(1):1–11, 2020.

[68] Charles X. Ling and Chenghui Li. Data mining for direct marketing: Problems and solutions. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, KDD'98, page 73–79. AAAI Press, 1998.

[69] H. Ling and D.W. Jacobs. Using the inner-distance for classification of articulated shapes. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 719–726 vol. 2, 2005. doi: 10.1109/CVPR.2005.362.

[70] Mohamed Loey, Ahmed ElSawy, and Mohamed Afify. Deep learning in plant diseases detection for agricultural crops: a survey. *International Journal of Service Science, Management, Engineering, and Technology (IJSSMET)*, 11(2):41–58, 2020.

[71] Changqing Luo, Jinlong Ji, Qianlong Wang, Xuhui Chen, and Pan Li. Channel state information prediction for 5G wireless communications: A deep learning approach. *IEEE Transactions on Network Science and Engineering*, 7(1):227–236, 2020. doi: 10.1109/TNSE.2018.2848960.

[72] Changqing Luo, Jinlong Ji, Qianlong Wang, Xuhui Chen, and Pan Li. Channel state information prediction for 5G wireless communications: A deep learning approach. *IEEE Transactions on Network Science and Engineering*, 7(1):227–236, 2020. doi: 10.1109/TNSE.2018.2848960.

[73] David JC MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

[74] Fan Meng, Peng Chen, Lenan Wu, and Xianbin Wang. Automatic modulation classification: A deep learning enabled approach. *IEEE Transactions on Vehicular Technology*, 67(11):10760–10772, 2018.

[75] Keiichi Mochida, Satoru Koda, Komaki Inoue, Takashi Hirayama, Shojiro Tanaka, Ryuei Nishii, and Farid Melgani. Computer vision-based phenotyping for improvement of plant productivity: a machine learning perspective. *GigaScience*, 8(1), 12 2018. doi: 10.1093/gigascience/giy153. URL `https://doi.org/10.1093/gigascience/giy153`. giy153.

[76] Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[77] Trishen Munisami, Mahess Ramsurn, Somveer Kishnah, and Sameerchand Pudaruth. Plant leaf recognition using shape features and colour histogram with k-nearest neighbour classifiers. *Procedia Computer Science*, 58:740–747, 2015. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2015.08.095. URL `https://www.sciencedirect.com/science/article/pii/S1877050915022061`. Second International Symposium on Computer Vision and the Internet (VisionNetâ15).

[78] João Camargo Neto, George E. Meyer, David D. Jones, and Ashok K. Samal. Plant species identification using elliptic fourier leaf shape analysis. *Computers and Electronics in Agriculture*, 50(2):121–134, 2006. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2005.09.004. URL https://www.sciencedirect.com/science/article/pii/S0168169905001560.

[79] Quang-Khue Nguyen, Thi-Lan Le, and Ngoc-Hai Pham. Leaf based plant identification system for android using surf features in combination with bag of words model and supervised learning. In *2013 International Conference on Advanced Technologies for Communications (ATC 2013)*, pages 404–407, 2013. doi: 10.1109/ATC.2013.6698145.

[80] Petr Novotný and Tomás Suk. Leaf recognition of woody species in central europe. *Biosystems Engineering*, 115(4):444–452, 2013. ISSN 1537-5110. doi: https://doi.org/10.1016/j.biosystemseng.2013.04.007. URL https://www.sciencedirect.com/science/article/pii/S1537511013000731.

[81] E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In *Neural Networks for Signal Processing VII. Proceedings of the 1997 IEEE Signal Processing Society Workshop*, pages 276–285, 1997. doi: 10.1109/NNSP.1997.622408.

[82] K Pankaja and G Thippeswamy. Survey on leaf recognization and classification. In *2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, pages 442–450, 2017. doi: 10.1109/ICIMIA.2017.7975654.

[83] Pornntiwa Pawara, Emmanuel Okafor, Lambert Schomaker, and Marco Wiering. Data augmentation for plant classification. In Jacques Blanc-Talon, Rudi Penne, Wilfried Philips, Dan Popescu, and Paul Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, pages 615–626, Cham, 2017. Springer International Publishing. ISBN 978-3-319-70353-4.

[84] Shengliang Peng, Hanyu Jiang, Huaxia Wang, Hathal Alwageed, Yu Zhou, Marjan Mazrouei Sebdani, and Yu-Dong Yao. Modulation classification based on signal constellation diagrams and deep learning. *IEEE transactions on neural networks and learning systems*, 30(3):718–727, 2018.

[85] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

[86] Nathan Price and Kurt Kosbar. Decoupling hardware and software concerns in aircraft telemetry sdr systems. In *International Telemetering Conference*, volume 54, pages 1–11, 2018.

[87] John G Proakis, Masoud Salehi, Ning Zhou, and Xiaofeng Li. *Communication systems engineering*, volume 2. Prentice Hall New Jersey, 1994.

[88] Anubha Pearline Sundara Sobitha Raj and Sathiesh Kumar Vajravelu. Ddla: dual deep learning architecture for classification of plant species. *IET Image Processing*, 13(12):2176–2182, 2019.

[89] Theodore S Rappaport et al. *Wireless communications: principles and practice*, volume 2. prentice hall PTR New Jersey, 1996.

[90] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4780–4789, Jul. 2019. doi: 10.1609/aaai.v33i01.33014780. URL `https://ojs.aaai.org/index.php/AAAI/article/view/4405`.

[91] Christian Reul, Martin Toepfer, and Frank Puppe. Cross dataset evaluation of feature extraction techniques for leaf classification. *International Journal of Artificial Intelligence Applications*, 7:01–20, 03 2016. doi: 10.5121/ijaia.2016.7201.

[92] Diego Rivera, Robert Allkin, Concepción Obón, Francisco Alcaraz, Rob Verpoorte, and Michael Heinrich. What is in a name? the need for accurate scientific nomenclature for plants. *Journal of Ethnopharmacology*, 152(3):393–402, 2014. ISSN 0378-8741. doi: https://doi.org/10.1016/j.jep.2013.12.022. URL `https://www.sciencedirect.com/science/article/pii/S0378874113009021`.

[93] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 430–443, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-33833-8.

[94] Stuart Russel, Peter Norvig, et al. *Artificial intelligence: a modern approach*. Pearson Education Limited London, 2013.

[95] Stuart Russel, Peter Norvig, et al. *Artificial intelligence: a modern approach*. Pearson Education Limited London, 2013.

[96] Muhammad Hammad Saleem, Johan Potgieter, and Khalid Mahmood Arif. Plant disease detection and classification by deep learning. *Plants*, 8(11), 2019. ISSN 2223-7747. doi: 10.3390/plants8110468. URL `https://www.mdpi.com/2223-7747/8/11/468`.

[97] Ramiro Sámano-Robles, Eduardo Tovar, João Cintra, and André Rocha. Wireless avionics intra-communications: Current trends and design issues. In *2016 Eleventh International Conference on Digital Information Management (ICDIM)*, pages 266–273. IEEE, 2016.

[98] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[99] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[100] Abhinav Sharma, Arpit Jain, Prateek Gupta, and Vinay Chowdary. Machine learning applications for precision agriculture: A comprehensive review. *IEEE Access*, 9: 4843–4873, 2021. doi: 10.1109/ACCESS.2020.3048415.

[101] Pramila P. Shinde and Seema Shah. A review of machine learning and deep learning applications. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–6, 2018. doi: 10.1109/ICCUBEA.2018.8697857.

[102] Jyoti Shirahatti, Rutuja Patil, and Pooja Akulwar. A survey paper on plant disease identification using machine learning approach. In *2018 3rd International Conference on Communication and Electronics Systems (ICCES)*, pages 1171–1174, 2018. doi: 10.1109/CESYS.2018.8723881.

[103] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[104] Krishna Singh, Indra Gupta, and Sangeeta Gupta. Svm-bdt pnn and fourier moment technique for classification of leaf shape. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 3(4):67–78, 2010.

[105] Bernard Sklar and FJ Harris. *Digital Communications: Fundamentals and Applications*. Prentice-hall, Englewood Cliffs, NJ, 1988.

[106] Oskar Söderkvist. Computer vision classification of leaves from Swedish trees. In *MS Thesis*. Linköping University, Linköping, Sweden, 2001.

[107] Oskar Söderkvist. Computer vision classification of leaves from Swedish trees. In *MS Thesis*. Linköping University, Linköping, Sweden, 2001.

[108] Mehran Soltani, Vahid Pourahmadi, Ali Mirzaei, and Hamid Sheikhzadeh. Deep learning-based channel estimation. *IEEE Communications Letters*, 23(4):652–655, 2019.

[109] Yupeng Song, Fazhi He, and Xiying Zhang. To identify tree species with highly similar leaves based on a novel attention mechanism for cnn. *IEEE Access*, 7: 163277–163286, 2019. doi: 10.1109/ACCESS.2019.2951607.

[110] Donald F Specht et al. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[111] Donald F Specht et al. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[112] Edney W Stacy. A generalization of the gamma distribution. *The Annals of mathematical statistics*, 33(3):1187–1192, 1962.

[113] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[114] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[115] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In Věra Kůrková, Yannis Manolopoulos, Barbara Hammer, Lazaros Iliadis, and Ilias Maglogiannis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2018*, pages 270–279, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01424-7.

[116] Jing wei Tan, Siow-Wee Chang, Sameem Abdul-Kareem, Hwa Jen Yap, and Kien-Thai Yong. Deep learning for plant species classification using leaf vein morphometric. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(1): 82–90, 2020. doi: 10.1109/TCBB.2018.2848653.

[117] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/tan19a.html`.

[118] Chin-Hung Teng, Yi-Ting Kuo, and Yung-Sheng Chen. Leaf segmentation, its 3d position estimation and leaf classification from a few images with very close viewpoints. In Mohamed Kamel and Aurélio Campilho, editors, *Image Analysis and Recognition*, pages 937–946, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-02611-9.

[119] KK Thyagharajan and I Kiruba Raji. A review of visual descriptors and classification techniques used in leaf species identification. *Archives of Computational Methods in Engineering*, 26(4):933–960, 2019.

[120] Transport Information Service. Climate/humidity table. `https://www.tis-gdv.de/tis_e/misc/klima.htm/`.

[121] Matthew A Turk and Alex P Pentland. Face recognition using eigenfaces. In *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 586–587. IEEE Computer Society, 1991.

[122] Muammer Turkoglu and Davut Hanbay. Recognition of plant leaves: An approach with hybrid features produced by dividing leaf images into two and four parts. *Applied Mathematics and Computation*, 352:1–14, 2019. ISSN 0096-3003. doi: https://doi.org/10.1016/j.amc.2019.01.054. URL https://www.sciencedirect.com/science/article/pii/S0096300319300694.

[123] Aalt Dirk Jan van Dijk, Gert Kootstra, Willem Kruijer, and Dick de Ridder. Machine learning in plant science and plant breeding. *iScience*, 24(1):101890, 2021. ISSN 2589-0042. doi: https://doi.org/10.1016/j.isci.2020.101890. URL https://www.sciencedirect.com/science/article/pii/S2589004220310877.

[124] Jana Wäldchen and Patrick Mäder. Plant species identification using computer vision techniques: A systematic literature review. *Archives of Computational Methods in Engineering*, 25(2):507–543, 2018.

[125] Jana Waldchen, Michael Rzanny, Marco Seeland, and Patrick Mader. Automated plant species identification–trends and future directions. *PLOS Computational Biology*, 14(4):1–19, 04 2018. doi: 10.1371/journal.pcbi.1005993. URL https://doi.org/10.1371/journal.pcbi.1005993.

[126] Li Wang, Stefano Fortunati, Maria S. Greco, and Fulvio Gini. Reinforcement learning-based waveform optimization for mimo multi-target detection. In *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, pages 1329–1333, 2018. doi: 10.1109/ACSSC.2018.8645304.

[127] Xiao-Feng Wang, Ji-Xiang Du, and Guo-Jun Zhang. Recognition of leaf images based on shape features using a hypersphere classifier. In De-Shuang Huang, Xiao-Ping Zhang, and Guang-Bin Huang, editors, *Advances in Intelligent Computing*, pages 87–96, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-31902-3.

[128] Xuan Wang, Weikang Du, Fangxia Guo, and Simin Hu. Leaf recognition based on elliptical half gabor and maximum gap local line direction pattern. *IEEE Access*, 8: 39175–39183, 2020. doi: 10.1109/ACCESS.2020.2976117.

[129] Chao-Kai Wen, Shi Jin, Kai-Kit Wong, Jung-Chieh Chen, and Pangan Ting. Channel estimation for massive mimo using gaussian-mixture bayesian learning. *IEEE Transactions on Wireless Communications*, 14(3):1356–1368, 2015. doi: 10.1109/TWC.2014.2365813.

[130] Chao-Kai Wen, Shi Jin, Kai-Kit Wong, Jung-Chieh Chen, and Pangan Ting. Channel estimation for massive mimo using gaussian-mixture bayesian learning. *IEEE Transactions on Wireless Communications*, 14(3):1356–1368, 2015. doi: 10.1109/TWC.2014.2365813.

[131] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010. doi: 10.25080/Majora-92bf1922-00a.

[132] Christoph Wick and Frank Puppe. Leaf identification using a deep convolutional neural network, 2017. URL `https://arxiv.org/abs/1712.00967`.

[133] Pengcheng Wu and Thomas G. Dietterich. Improving svm accuracy by training on auxiliary data sources. ICML '04, page 110, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015436. URL `https://doi.org/10.1145/1015330.1015436`.

[134] Stephen Gang Wu, Forrest Sheng Bao, Eric You Xu, Yu-Xuan Wang, Yi-Fan Chang, and Qiao-Liang Xiang. A leaf recognition algorithm for plant classification using probabilistic neural network. In *2007 IEEE International Symposium on Signal Processing and Information Technology*, pages 11–16, 2007. doi: 10.1109/ISSPIT.2007.4458016.

[135] Stephen Gang Wu, Forrest Sheng Bao, Eric You Xu, Yu-Xuan Wang, Yi-Fan Chang, and Qiao-Liang Xiang. A leaf recognition algorithm for plant classification using probabilistic neural network. In *2007 IEEE International Symposium on Signal Processing and Information Technology*, pages 11–16, 2007. doi: 10.1109/ISSPIT.2007.4458016.

[136] Yan-Hao Wu, Li Shang, Zhi-Kai Huang, Gang Wang, and Xiao-Ping Zhang. Convolutional neural network application on leaf classification. In De-Shuang Huang, Vitoantonio Bevilacqua, and Prashan Premaratne, editors, *Intelligent Computing Theories and Application*, pages 12–17, Cham, 2016. Springer International Publishing. ISBN 978-3-319-42291-6.

[137] Zhangnan Wu, Yajun Chen, Bo Zhao, Xiaobing Kang, and Yuanyuan Ding. Review of weed detection methods based on computer vision. *Sensors*, 21(11), 2021. ISSN 1424-8220. doi: 10.3390/s21113647. URL `https://www.mdpi.com/1424-8220/21/11/3647`.

[138] Hao Ye, Geoffrey Ye Li, and Biing-Hwang Juang. Power of deep learning for channel estimation and signal detection in ofdm systems. *IEEE Wireless Communications Letters*, 7(1):114–117, 2017.

[139] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 818–833, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10590-1.

[140] Xinchuan Zeng and Tony R. Martinez. Distribution-balanced stratified cross-validation for accuracy estimation. *Journal of Experimental & Theoretical Artificial Intelligence*, 12(1):1–12, 2000. doi: 10.1080/095281300146272. URL `https://doi.org/10.1080/095281300146272`.

[141] Zalikha Zulkifli, Puteh Saad, and Itaza Afiani Mohtar. Plant leaf identification using moment invariants  general regression neural network. In *2011 11th International Conference on Hybrid Intelligent Systems (HIS)*, pages 430–435, 2011. doi: 10.1109/HIS.2011.6122144.

# VITA

Viraj Kishorkumar Gajjar received a Bachelor of Engineering in Electronics and Communications Engineering in July 2014 from the Indus Institute of Technology and Engineering (Gujarat Technological University), Ahmedabad, Gujarat, India. He taught coursework as a graduate teaching assistant in the electrical engineering department at Missouri University of Science and Technology under Dr. Kurt Kosbar. He received his Doctor of Philosophy in Electrical Engineering at Missouri University of Science and Technology in July 2022. His research interests included digital communications, digital signal processing, statistical signal analysis, computer vision, machine learning, and deep learning.