CNN-BASED PLANT SPECIES CATEGORIZATION USING NATURAL IMAGES

A DISSERTATION SUBMITTED TO THE GRADUATE DIVISION OF THE UNIVERSITY OF HAWAIʻI AT MĀNOA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE

APRIL 2020

By

Jonas Krause

Dissertation Committee:

Kyungim Baek, Chairperson
Lipyeow Lim, Chairperson
Edoardo S. Biagioni
Curtis C. Daehler
Peter Sadowski

Keywords: Plant Species Categorization, Convolutional Neural Networks, Guided Multi-Scale Analysis, *WTPlant* System, Integration of Domain-Specific Knowledge.

ABSTRACT

Automatic identification of plants from natural images is a challenging problem that is relevant to both the disciplines of Botany and Computer Science. The classification of plant images at the species level is a computer vision task called fine-grained categorization. This categorization problem is particularly complicated due to a large number of plant species, the inter-species similarity, the large-scale variation in appearance, and the lack of annotated data. Despite the availability of dozens of plant identification mobile applications, categorizing plant species from natural images remains an unsolved problem – e.g., most of the existing applications do not address the multi-scale nature of this type of image. Furthermore, an automated system capable of addressing the complexity of this computer vision problem has important implications for society at large, not only in preserving ecosystem biodiversity and public education but also in numerous agricultural activities such as detecting abnormalities in plants and analyzing food crops.

In this dissertation, I present a new approach to the problem of automatically categorizing plant species using photos taken in nature. Essentially, this approach assembles a collection of Convolutional Neural Networks (CNN-based) to create a plant categorization system that I named *WTPlant* (*What's That Plant?*). One of the novelties of this system is a preprocessing method that extracts multi-scale samples from natural images, making the classification models more robust to variations in the scale of the plant. A comprehensive experimental evaluation of this new preprocessing method compares its performance with frequently used data augmentation techniques over different classification models of the system. *WTPlant* also enables the categorization of multiple plant components simultaneously by employing distinct classification pipelines for plants (leaves, branches, bushes, and trees) and flowers. The combination of these multi-organ analyses ensures a broader categorization process. It can be further extended by adding pipelines for fruits, barks, roots, etc., depending on the availability of annotated images. In summary, this new approach locates multiple plant organs in a natural image and guides the extraction of representative samples at various scales used to train and test state-of-the-art CNN classification models.

To apply the *WTPlant* system in a real-world environment, I implement a scale-up process that adapts the classification models. In this process, models have their top classification layers replaced to accommodate a more significant number of plant species. But due to a lack of training data, these models have to be pre-trained to achieve satisfactory performance. As a result, I also implement the integration of domain-specific knowledge to create plant and flower expert classification models. Initially focusing on the University of Hawai'i Mānoa campus plants, this research aims to produce the most accurate system for classifying Hawaiian plants and make it available to botanists, tourists, and the entire community to use. As a case study, I create a mobile version of the *WTPlant* system to categorize plant species from the Harold L. Lyon Arboretum, a University of Hawai'i Research Unit located at the upper end of the Mānoa Valley.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Traditionally, botanists analyze different aspects of a plant to identify its species. Focusing only on visible characteristics, the correct categorization of plants requires considerable knowledge [72]. Unsurprisingly, some species have specific traits that need to be considered for the correct plant categorization. As an example, some plants can be distinguished from very similar species only based on their seeds or pods. Due to particular characteristics like these, it is almost impossible for the general public and challenging even for botanists to identify a plant species from a single image correctly.

Knowledge of plant species is essential to protect the biodiversity of any flora, and an automated system to identify plants has important applications ranging from conservation to agriculture. For conservation purposes, an automated system can capture different phenomena throughout the plant's life cycles like germination, budding, and flowering. Previous methods to extract this information from natural images are incredibly tedious. For agriculture, most applications are related to automatic food crop analysis, especially the identification of pests, diseases, and invasive species. The improvement in these agricultural efforts can, in turn, lead to better crop control and management, higher-yielding food production, and possibly a reduction in pesticide use.

Over the last few years, Machine Learning (ML) approaches have shown promising results in many computer vision problems, including plant identification. Previous efforts have used hand-designed features of leaves, flowers, and fruits [6, 10, 18, 37, 72], and most of them are restricted to the analysis of fairly controlled images with clean backgrounds. However, categorizing plant species relying on morphological characteristics extracted from well-controlled images is quite different from the noisy natural images found in real-world classification problems.

The categorization of natural images can be extremely challenging due to complex backgrounds, different illumination sources, occlusions, shadows, and objects appearing in any scale. Because of these factors, the automated analysis of natural scenes is a complex task for computer vision systems. This problem is further exacerbated by the necessity of using unconstrained images, varying in size, resolution, scale, and focus. While the human visual processing system navigates those factors with ease, an equivalent computational model for plant identification using natural images is still an open problem.

More recently, Deep Learning (DL) methods have been introduced to this task [1, 5, 38, 41, 52, 63, 67, 70] driven by the success of Convolutional Neural Networks (CNNs). These deep convolutional approaches have been a growing trend in the computer vision field, demonstrating impressive results in various tasks involving natural images. The plant categorization system (*WTPlant*) described in this dissertation utilizes these DL methods, further extending it with the use of multiple stages and making different CNN models work together in a single framework.

1.1 Problem Statement

A real-world plant identification application has to deal with natural images, which is a major challenge for computer vision systems. In this dissertation, the plant categorization problem is stated as the analysis of an unconstrained natural image of a plant to identify its species, and is defined as: given a natural image of a plant;

- Identify the presence and location of multiple plant organs;
- Define the most representative areas of the image for this categorization task;
- Analyze plants independent of position, occlusion, scale, or background;
- Improve the training process of cutting edge computer vision methods for best categorization accuracy.

I address these issues by:

- Segmenting the plant organs from a complex background with scene parsing approaches;
- Calculating bounding boxes over the segmented areas;
- Extracting guided multi-scale representative samples for analysis at various scales, avoiding partial occlusions, and eliminating non-plant background objects;
- Implementing a new multi-scale classification process that, combined with the guided multi-scale data augmentation, makes the *WTPlant* system more robust to scale variations;
- Integrating knowledge from domain-specific datasets to create expert CNN models and improve categorization accuracy.

As a brief overview, this novel approach to categorize plants using natural images implements multiple classification pipelines (e.g., plants and flowers) and different CNN models working together to guide the extraction of discriminatory scale-invariant features. Each pipeline uses different areas of the query image identified as plant or flower by the initial scene parsing method. *WTPlant* processes these regions of interest into samples of different scales and classifies them using state-of-the-art CNN models. Then I combine the results from each of these classification pipelines to obtain more accurate predictions in a process reminiscent of ensemble techniques, outputting the final predicted plant species.

In contrast to current plant identification methods that use hand-designed features or simple CNN models, *WTPlant* is carefully designed to handle natural images and perform a multi-scale classification of different organs of the analyzed plant. Moreover, I developed this plant categorization system focusing on the following research questions:

[**Question 1**] Where are the most representative areas in the image for the plant categorization?

To address this first question, I explore scene parsing approaches that implement CNNs to locate multiple objects in the image. These models are previously trained using annotated scenes with the most common indoor and outdoor objects. In this way, the initial scene parsing analysis delimits regions of interest for each of the detected objects, including plants and flowers. This dissertation describes how these segmented regions are determined and defines square bounding boxes delimiting these areas for the plant categorization task. The difficulty of correctly defining the specific region of a plant or flower (especially when other plants are also present in the image) makes the guided bounding boxes a good strategy to collect samples for the classification models. This approach showed satisfactory initial results and is incorporated into the *WTPlant* system, creating the first contribution of this dissertation. In a short answer, I take advantage of a pre-trained CNN for the scene parsing of a natural image to locate plant and flower regions and use this information to delimit the most representative areas for this task.

[**Question 2**] How to classify plants and flowers at different scales?

After defining the bounding boxes of the located plants and flowers, the *WTPlant* system implements a preprocessing method to provide a multi-scale analysis of the selected areas. The largest connected plant and flower areas guide the extraction of samples at various scales from the dominant plant species in the image. Implementing this approach, I create a new guided multi-scale data augmentation process. This new data augmentation process is the second contribution of this dissertation, and I developed it to make the system capable of classifying plants and flowers at different scales. It is a novel approach to the multi-scale analysis of plants and aims to make the classification models of the *WTPlant* system more robust to scale variations. Experiments detailed in this dissertation support this hypothesis and show how CNNs trained using this new data augmentation approach outperform similar models trained by commonly used methods such as resizing and random crop.

[**Question 3**] How to improve the classification process of the plant categorization system?

To answer this question, I present an empirical analysis of the multi-scale methodology. First, the multi-scale analysis is implemented during the classification process of all unseen images. Different from other methods, *WTPlant* creates this multi-scale analysis of new images during their categorization process. This process aims to analyze various scales of plant and flower areas from the same target image. It combines the classification results of different organs of the plant in multiple scales to a better categorization of a single image. Secondly, the *WTPlant* system has its performance improved by pre-selecting scales and using their mirrored images. Both approaches address this third research question and improve the classification process of state-of-the-art CNNs. Consequently, I answer this question by implementing a multi-scale classification process for the analysis of the same image at various scales, and a pre-selecting process to use the most appropriate scales for different classification problems (plants and flowers).

[**Question 4**] How to expand the plant categorization scope while maintaining high accuracy? For the deployment of the *WTPlant* system in larger scenarios, I expand its initial scope by replacing the top layers of previously trained classification models to accommodate the number of plant species from the new environment. This scope expanding process requires the retraining of both plant and flower classification models of the system. So gathering together a representative group of plant images from the target plant species is necessary to expand the categorization scope. Furthermore, the collection and revision of a representative dataset is a necessary step to assure that the classification models learn the discriminative features existing between the plant species. During this procedure, it is important to consider the presence of plant species that may be impossible to distinguish using only visual features. Consequently, I recommend the assistance of a botanist to the creation and correct annotation of new datasets. Taking into account the unique characteristics of each plant species in different regions of the world, I prepare the newly collected dataset and adapt classification models to this new scope, expanding the plant categorization system to a broader environment.

An adaptation on the top classification layers enables previously trained CNNs to work on a significantly larger number of species but does not guarantee a high categorization accuracy. To assist in the training of CNNs over the dataset with an expanded scope, I create plant and flower expert models to help the fine-tuning of the *WTPlant* classification models. This is a problematic step, mostly due to the demanding computational effort required to train these models over multiple and massive datasets. Therefore, I utilize the integration of domain-specific knowledge from different plant-related datasets to keep a high accuracy when expanding the plant categorization scope. Also called fine-tuning, these repeated training processes cluster domain-specific knowledge starting from general well-trained models to create plant and flower expert ones. The fine-tuned CNNs take much longer to be trained but present more accurate results throughout the experiments.

1.2 Contributions

The contributions of this dissertation are:

1. A localization method using a pre-trained CNN for the scene parsing of a natural image and the definition of the most representative areas of the image for the plant categorization task;

2. A guided multi-scale data augmentation process implemented to make CNNs more robust to scale variations when analyzing plants in natural images;

3. The *WTPlant* system with an expandable framework to a broader analysis of multiple plant organs;

4. A multi-scale classification process customizable to different organs of the plant and with distinct CNN models for comparative analysis;

5. Comprehensive experimental validation and evaluation of the designed system on different datasets containing natural images, resulting in a considerable improvement in accuracy when multi-scale approaches are used;

6. A suitable approach to expand the scope of the *WTPlant* and cover all the species from a specific environment;

7. The integration of domain-specific knowledge to create plant and flower expert models;

8. The publication of pre-trained weights[1] from plant and flower expert CNNs to assist other plant categorization methods on the training of their classification models;

9. A mobile application version of the *WTPlant* system that can be adapted to multiple environments and help spread knowledge of Hawaiian flora and culture;

1.3 Publications

The following publications are related to this research. I present them chronologically, showing the evolution of the *WTPlant* system and improvements performed in the guided multi-scale approach throughout the past years.

- Jonas Krause, Gavin Sugita, Kyungim Baek, and Lipyeow Lim. WTPlant (What's That Plant?): A Deep Learning System for Identifying Plants in Natural Images. In *Proceedings of the International Conference on Multimedia Retrieval (ICMR 2018)*. ACM Press, 2018.

- Jonas Krause, Gavin Sugita, Kyungim Baek, and Lipyeow Lim. What's That Plant? WTPlant is a Deep Learning System to Identify Plants in Natural Images. In *BMVC Workshop on Computer Vision Problems in Plant Phenotyping (CVPPP 2018)*. BMVA Press, 2018.

- Jonas Krause, Kyungim Baek, and Lipyeow Lim. A Guided Multi-Scale Categorization of Plant Species in Natural Images. In *CVPR Workshop on Computer Vision Problems in Plant Phenotyping (CVPPP 2019)*. IEEE Press, 2019.

[1] https://github.com/jonaskrause/Plant_Flower-Expert_CNN_Models

1.4 Dissertation Outline

The structure of the dissertation is organized as follows. In Chapter 2, I briefly introduce the existing approaches and applications designed for the plant categorization problem listing them according to their feature extraction approach (hand-designed features in Section 2.1.1 and deep learning in Section 2.1.2). The most used deep learning models (CNN) and its variations are presented in Section 2.2. Chapter 3 presents the initial steps of a plant categorization system by describing how to define the most representative areas of a natural image for this task. Addressing the multi-scale issue, Chapter 4 presents an overview of the *WTPlant* system describing its framework (Section 4.1), the guided multi-scale approach implemented as preprocessing stage (Section 4.2), the improvement of plant and flower classification processes (Section 4.3), the addition of new and pre-trained CNN models (Section 4.4), and the Graphical User Interface (Section 4.5). Chapter 5 extends the classification models of the *WTPlant* by increasing the number of analyzed plant species and using massive datasets to create domain-specific models. In Chapter 6, I combine plant and flower predictions and improve the confidence analysis of the system to output the final categorized species. Chapter 7 presents a case study, where I deploy the *WTPlant* system to categorize plant species present at the Harold L. Lyon Arboretum. The last chapter describes the conclusions of this dissertation, future work, and possible real-world applications of the developed system in botany and agriculture.

CHAPTER 2
RELATED WORK

In this section, I survey previous works relevant to the identification of plants in the field of computer vision. I begin by briefly introducing the different existing applications and how some of them have used DL approaches to solve this problem.

2.1 The Plant World

Biodiversity Informatics appeared in the 1990s as a multi-disciplinary field on the frontier of Computer Science and Taxonomy [27]. Since then, numerous approaches have been used in the attempt to automatically handle these taxa by organizing, accessing, visualizing, and analyzing biodiversity data. Particularly for plants, researchers have been using hand-designed feature methods since the beginning. These methods consist of manually extracting discriminative information from sample images. As an example, in a grayscale image, a simple edge detection algorithm works by finding areas of the image that suddenly change in intensity. A large number of hand-designed features have been used for plant identification [10, 26, 37, 39], such as edge features, shape properties, Kernel Descriptors (KDES), Scale-Invariant Feature Transform (SIFT), and others.

Recently, Wäldchen and Mäder [72] presented a systematic review to analyze published papers that produce an automated plant identification system. This review focuses on the image acquisition and the feature extraction steps of a generic image-based plant classification method. They list 120 papers that use only hand-designed features, showing the relationship between each extracted feature and the identification factor analyzed in the plants. Not surprisingly, almost 90% of the reviewed papers consider only leaves on their methods and rely mainly on shape identification factors to classify the plant species correctly. As a result, datasets are created by posing each leaf in plain background and taking an individual picture. Morphological features of leaves have been used for the past decades and yielded functional applications such as the *LeafSnap* [37]. However, they are not suitable for the analysis of natural images and require particular leaf samples to work correctly.

Other reviewed papers focused on flower image classification, redirecting the feature extraction from morphological features to textural ones. Approaches that rely only on flower images to identify plant species are not very common due to their seasonality. This process also indicates how hard it is to hand-design these manually extracted features when using multiple plant organs. Still, it suggests that a framework combining individual leaf and flower analyses may result in a more robust approach for the plant species categorization problem. The following sections detail the existing methods for categorizing a plant image, dividing them into DL and non-DL approaches.

2.1.1 Non-Deep Learning Approaches

Two of the most famous plant identification applications use hand-designed features. *LeafSnap* works as described above, by using images of leaves on a plain background. And the other one (*Folia*) works by implementing segmentation methods on natural leaf images and then extracting hand-designed features.

LeafSnap

LeafSnap is a famous (if not the most popular) mobile application for leaf image classification. It all started in 2003 with computer science professors from Columbia University and the University of Maryland who wanted to apply facial recognition software to the natural world. However, facing the difficulty of analyzing natural images, they introduced the idea of taking a photo of a leaf isolated on a solid light-colored background to facilitate shape discrimination. Figure 2.1 shows an example of such leaf images photographed in laboratory [37].

Figure 2.1: *Halesia tetraptera* leaf photographed in laboratory, source: *http://www.leafsnap.com/*.

Presented by Kumar *et al.* [37] in a more recent paper, *LeafSnap* is described as the first framework created to classify plant species using automatic vision recognition methods. The description of this end-to-end application details the process of classifying new leaf images among 185 tree species. This system relies mainly on hand-designed features for classification, but other computer vision techniques are also applied. For example, in the first stage of the framework, spatial envelope properties and Support Vector Machine (SVM) are used to determine what is leaf and what is not. After that, saturation-based segmentation methods are implemented, and curvature-based shape features of the leaf's contour over multiple scales are extracted from the segmented area. The final classification is done by the k-nearest neighbors (k-NN) algorithm using the set of features called Histograms of Curvature over Scale (HoCS). An impressive characteristic of this application is that it saves the Global Positioning System (GPS) coordinates and timestamp of each photo taken, hoping to be able to map the biodiversity of a region over space and time. The limitation, however, is that this system requires a single leaf specimen to be photographed and, even using field images on their training datasets, *LeafSnap* is not designed to analyze natural images.

Folia

This application focuses on extracting hand-designed features to segment and classify leaves using natural images. Cerutti *et al.* [10] present this interesting framework called *Folia* aiming to analyze the same leaf features that botanists use to classify tree species. They include leaf size, global shape, venation, basal and apical shapes, type of margins, number of lobes, and others. For this application, the authors focused on non-compound simple leaf images with several lobes centered and vertically-oriented. Only 50 different species are researched, which may account for their excellent results when compared with other non-DL methods. They also exclude the analysis of compound leaves, restricting their method to simple, centered, vertically oriented, and palmately lobed leaf images. As an example, Figure 2.2 presents two natural images from the *Folia* application interface. Even with all these restrictions, *Folia* may be the framework that best targets a leaf in natural images. Nonetheless, segmenting plants and their leaves from natural images is, by itself, a big challenge. Therefore, new methods such as DL models trained for segmentation are also yielding satisfactory results.

2.1.2 Deep Learning Approaches

Historically, the concept of DL originated from artificial neural network research. In 2006, an efficient learning algorithm to train deep networks was introduced [23, 22] and DL became one of the key research areas in ML. The essential quality of DL models is the presence of several processing layers in their neural networks. These layers are hierarchically organized to learn deep sophisticated features by progressively following simpler ones. Different techniques on how to process these layers create different DL models. But a shared property in all models is that each step into their deep

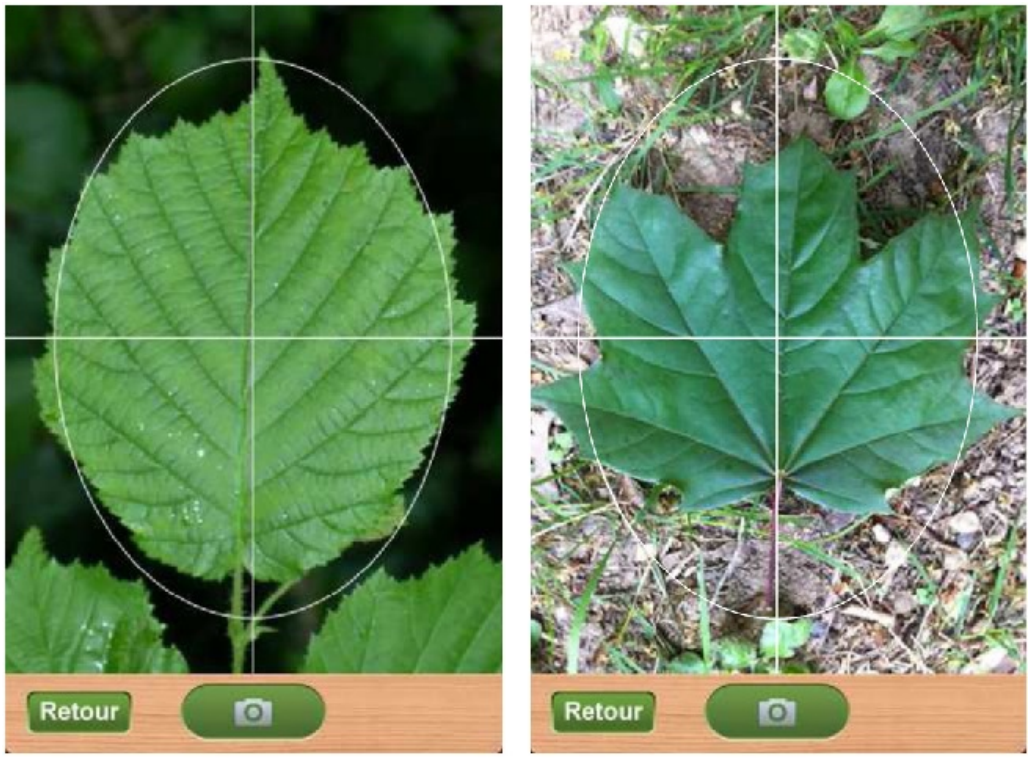


Figure 2.2: Two natural images (simple and palmately lobed leaves) from *Folia* [10] application.

architecture creates a more abstract representation of the data. Thereby, DL models create features based upon the training data and discover hierarchical dependencies in the analyzed dataset. The novelty of data-driven features is the main strength of DL methods and is revolutionizing the ML field.

Currently, different DL models are being used to address the plant identification problem, and they are listed below. Most of them are well-known CNN models adapted to work with plant images. However, few of them present new approaches designed to address specific aspects of the plant identification problem using natural images.

Pl@ntNet

Pl@ntNet is a world-scale participatory platform and information system dedicated to the monitoring of plant biodiversity through image-based plant identification [18] and has recently migrated its classification approach from hand-designed features classifiers to DL methods. This project started in 2010 and has evolved with iterative developments based on multimedia information retrieval, data aggregation, and integration by a growing community of volunteers [28, 29]. Nevertheless, a considerable improvement in the plant recognition performance was only observed in 2015 when a CNN was introduced in their classification process [1].

Using the *GoogLeNet* [65] CNN architecture, *Pl@ntNet* fine-tunes its model periodically using new observations with verified annotated species. This CNN model implements inception modules to improve the multi-scale analysis. These modules are described in detail in section 2.2.2. The main advantage of this application is that it collects 2.5 millions of images from around the world to train its classification models. *Pl@ntNet* is still expanding to cover all North American plant species.

LeafNet

Lab produced leaf images (as presented in Figure 2.1) are also used in DL frameworks to classify plants. Implementing a simple CNN model, Barré *et al.* [5] present the application called *LeafNet* and compare their results with *LeafSnap* performance. They used three datasets (*LeafSnap*, *Flavia*, and *Foliage*) to train and test their CNN-based plant identification system. However, their method is restricted to 185 plant species listed by *LeafSnap* dataset and the plant has to be manually prepared to be photographed on a white background, similar to *LeafSnap*. *LeafNet* also downscale all images to a fixed size of 256x256 pixels with no segmentation to train their CNN model, which makes the train and test images to lose important discriminative information.

Even so, satisfactory results are reported on all three datasets. By comparing CNN models with hand-designed feature methods such as the *LeafSnap* system, Barré *et al.* [5] showed that learning features by using a CNN provides a better representation of leaves and consequently better discrimination. An interesting point of the *LeafNet* system is that it is entirely available online[1] and released under a free software license. Therefore, other researchers can download it and train the *LeafNet* framework on different datasets or even collaborate with this ongoing research.

Deconvolutional AlexNet

Adapting a CNN model called *AlexNet* [36] and using a dataset of 44 plant species, Lee *et al.* [41] focus on the classification of preprocessed leaf images. From these lab-produced images, leaves are segmented by extracting the foreground pixels using the HSV (Hue, Saturation, and Lightness) color space information. Their first experiment uses a pre-trained CNN and fine-tunes it using the segmented leaves. Initial results are not as expected, so they decided to create a deconvolutional structure to verify what features the CNN learned visually. During this process, they noticed that the trained model is focusing almost exclusively on the contour and shape of leaves. Due to low accuracy results, they conclude that leaf shape is not a good choice to identify plants, which is not necessarily true. Morphological features of leaves have been heavily and successfully used for plant species categorization. In this case, non-satisfactory results may be a consequence of a poor segmentation process that creates misleading training data for the CNN to learn.

[1] https://leafnet.pbarre.de/

In a second attempt, each leaf image is manually cropped into samples within the area of the leaf. Fine-tuning the same pre-trained CNN with these new samples and using the deconvolutional structure to observe the transformation of the features layer by layer, they are able to correctly classify most of the test images based on the venation of the leaves. Producing good results over a limited dataset, Lee *et al.* [41] present an excellent alternative to the initial approach of taking photos of leaves with controlled backgrounds. Using these manually segmented samples, they forced their method to discover other discriminative features, in this case, the structures of leaf veins.

Hourglass CNN

There is a particular demand to quantify images of food crops accurately. Analyses of these type of images generally focus on one species to produce higher-yielding plants. For cereal plants, this analysis is measured in terms of grains present on spikes at the tip of the plant. Using their dataset of wheat plant images, Pound *et al.* [52] implement a CNN in an hourglass architecture to count wheat grains. The presented model is very similar to the previous one, where a deconvolutional structure is used to decode the extracted features and, in this case, pinpoint the location of the grains. The difference is that the hourglass CNN model has more connections between the layers, which improves the deconvolutional functionality.

The proposed method seems to be very effective, reporting high accuracy results on the grains counting process. However, the dataset used is called ACID (Annotated Crop Image Dataset), which contains a limited number of 520 images. Because of that, they had to implement several data augmentation strategies and test multiple CNNs with different input image resolutions to obtain satisfactory results. The main limitation of this method is this dataset, which is also using lab prepared images with dark backgrounds. Therefore, this approach needs classification models trained to work with natural images. Despite that, the approach of preprocessing training images by cropping the spike areas and resizing these samples to be inputted in different hourglass CNNs works very well. As a result, their CNNs can concentrate on discriminative features specific to the localization of the grains.

Customized Residual Network

Sun *et al.* [63] present one of the few papers that address the classification of natural images of entire plants and trees. The proposed plant classification problem includes 100 plant species with high-resolution images of individual bushes and trees. These images are collected from the Beijing Forestry University campus, and they are available online in the BJFU100[2] dataset. Figure 2.3 presents some of them. In these images, it is more evident how challenging the classification of natural images is. They present a variety of backgrounds, different illumination focuses, shadows, and it is not always possible to identify a leaf of the plant.

[2]https://pan.baidu.com/s/1jILsypS

Figure 2.3: Example images of trees and bushes of the BJFU100 [63] dataset.

Knowing these obstacles, they implement a modified version of the Residual Network [20] (ResNet) to classify these images. The implemented residual blocks aim to extract even deeper discriminative features by adding the previous input layer with the last extracted features. In their model, a pre-trained ResNet works as a bottleneck structure between an initial convolutional block and the last layers of the network. Like this, they adapted the ResNet architecture to their needs, customizing this successful CNN model and fine-tuning it with their dataset. Nevertheless, the poor preprocessing of the high-resolution images probably impacted their method negatively. Their preprocess consists only of the downsizing of the original 3120x4208 pixel resolution images to fit the first convolutional layer of their CNN that receives a 224x224 area as the input image. So, drastically reducing the size of an image will inevitably make it lose a lot of relevant information and, in this case, also change the plant aspect ratio.

2.2 Convolutional Neural Networks

Nowadays, many computer vision tasks rely heavily on using Convolutional Neural Networks (CNNs). While the shape of the data is often ignored in the traditional neural networks by flattening the input image into a 1-dimensional array, CNNs keep the structure of the data by using 3-dimensional activations. It is one of the reasons why CNNs are more suitable for tasks in computer vision since neighboring pixels in an image tend to have similar values, and the values in RGB (Red, Green, and Blue) channels of a pixel are closely related. In this way, these biologically-inspired computational models try to emulate human vision behavior by detecting local features such as edges, curves, and shapes. The idea of neuronal cells in the visual cortex recognizing specific characteristics is the basis behind CNNs.

A simple version of a CNN consists of a sequence of convolutional blocks. For each block, the first layer is a convolutional one, which can be understood as a scanning layer. During the training process, this layer learns a predefined number of filters, which produce the discriminative feature maps. The second layer is an activation one, generally implemented using the rectified linear unit (ReLU) function and responsible for adding non-linearity to the model. The last layer of a convolutional block is a pooling layer, a simple subsampling discretization process for dimensionality reduction. The final block of a CNN is composed of fully connected layers, where all neurons of the next layer are connected to every neuron of the previous layer. The last layer of this block and the entire network is the output, containing one neuron for each class of the classification problem. This last layer is also fully connected to the previous one and generally implements a softmax function. Figure 2.4 illustrates a general CNN architecture.

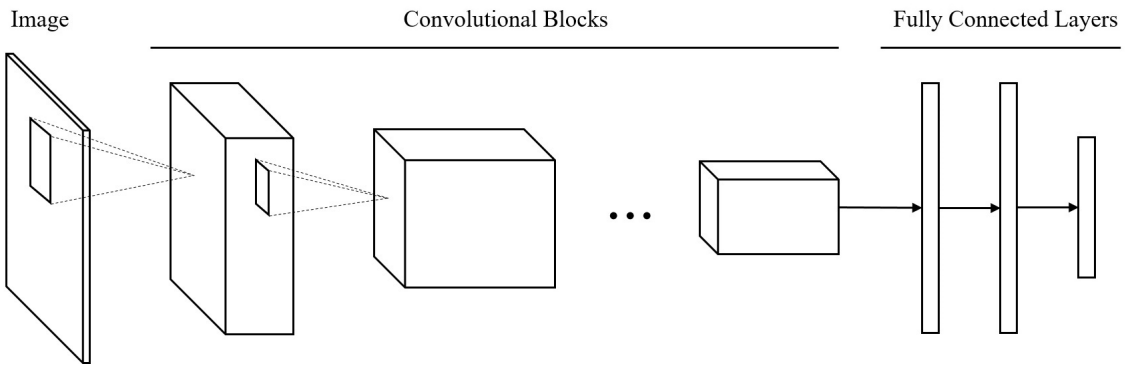


Figure 2.4: A common CNN architecture.

One of the most famous CNN architectures is the *AlexNet* [36]. This network has been widely cited since its creation in 2012, and it can be considered as one of the most influential publications in the field. Presented in Figure 2.5, this CNN architecture receives a 224x224 pixel color image as input data. It extracts features through five convolutional blocks with pre-designed widths, heights, depths, convolutional windows sizes, and other specific parameters. The last convolutional block passes the extracted features to the fully connected layers, which classifies them among the 1000 output classes. *AlexNet* is a well-studied CNN architecture and is designed for large-scale general datasets. As a result, this CNN model commonly outperforms hand-designed methods presenting impressive results in several visual challenge campaigns.

2.2.1 Residual Networks

In 2015, a new concept called residual blocks (or Residual Neural Networks - *ResNets*) was presented by He *et al.* [20], which allows training of much deeper CNNs. They are designed to address the degradation problem that appears when very deep models' accuracy gets saturated. To overcome this issue, residual blocks allow a deeper exploration of features by introducing *skip*

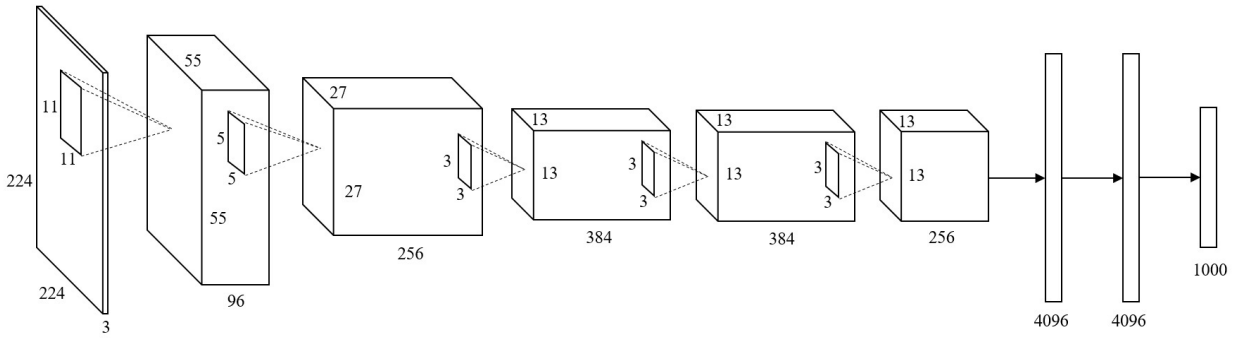


Figure 2.5: Simplified view of the *AlexNet* architecture [36].

or *shortcut* connections – the input passes through the convolutional blocks, and it is added to the output after two layers. A *ResNet* with over 100 layers won the ImageNet challenge in 2015 [58]. The intuition that deeper networks extract even more discriminating features makes *ResNet* models good candidates for dealing with fine-grained categorization problems. Figure 2.6 shows the implementation of the residual block.

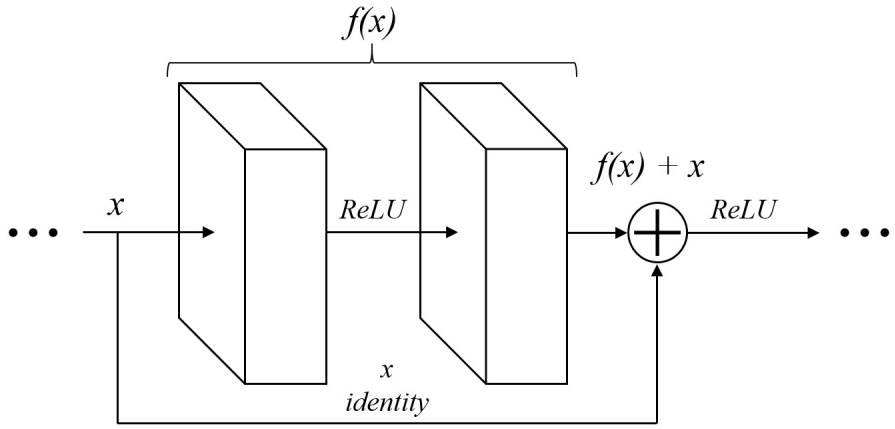


Figure 2.6: Residual block architecture introduced by He *et al.* [20].

As an example, Figure 2.7 presents one of the implemented *ResNets* with 18 layers. The inputted image goes through an initial convolutional block of 64 filters that reduces its dimensionality by half during the pooling process. After this initial process, it enters on two identical residual blocks and again is reduced by half (dotted lines) on the next residual structure with 128 filters per convolutional block. This process is repeated extracting more and more feature maps and finishes with a fully connected layer that will perform the classification among the 1000 classes.

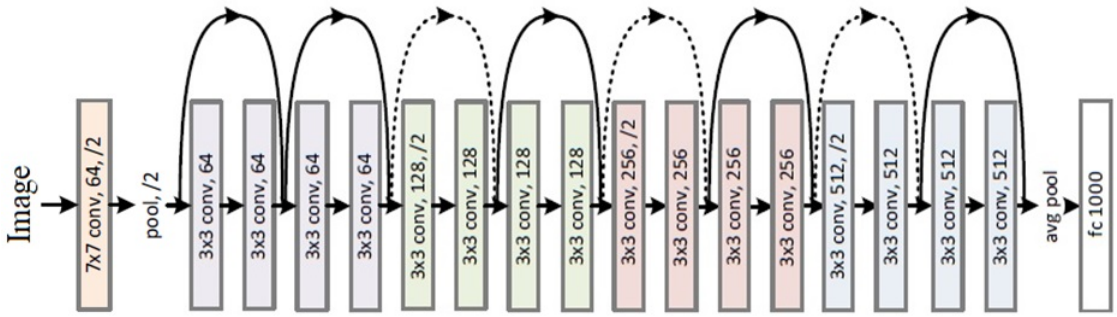


Figure 2.7: Simplified view of the *ResNet18* architecture [20].

2.2.2 Inception Module Networks

The latest CNN models have been exploring the concept of Inception Modules. Introduced by Szegedy *et al.* [65], inception modules address the multi-scale issue by implementing multiple convolutional filters of various sizes in parallel. The objective of these multiple filters is to identify the object's salient parts, even with a significant variation in size. It also uses 1x1 convolutional filters to reduce dimensionality before expensive convolution operations. As a simplified example, Figure 2.8 shows an inception module with three different filter sizes (5x5, 3x3, and 1x1) as well as the dimensionality reduction performed by an average pooling and a 1x1 convolutional filter.

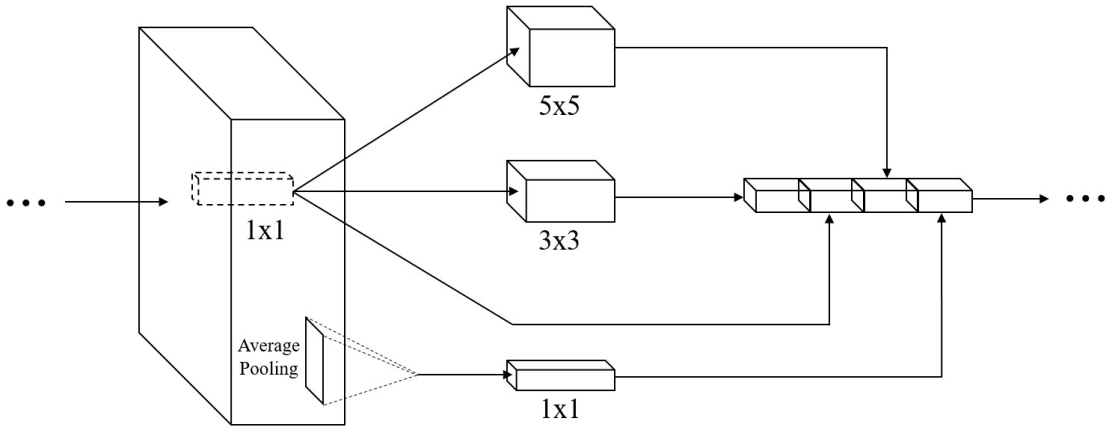


Figure 2.8: Inception Module introduced by Szegedy *et al.* [65].

The use of inception modules seeks to create a wider network instead of a deeper one, making it easier to train. As an example, Figure 2.9 shows the architecture of an inception module network called *GoogLeNet* where each highlighted dotted red box represents an inception module. This is one of the first CNNs that moved away from the common approach of stacking convolutional and pooling layers on top of each other in a sequential architecture. This model won the ImageNet competition in 2014 [58] and has proved to be a very efficient DL strategy.

Presenting top accuracy results on numerous computer vision challenges, *GoogLeNet* supports the idea that approximated sparse structures may efficiently represent dense building blocks and improve the architectures of CNNs. By reducing the computational effort when dealing with sparse structures and implementing different filter sizes for the multi-scale issue presented in most computer vision problems, inception networks indicate that switching to more scattered models may be a useful idea in general.

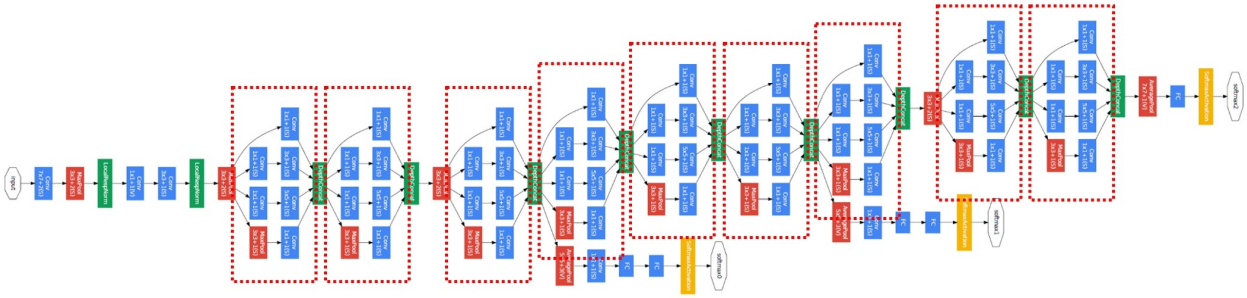


Figure 2.9: Overview of the *GoogLeNet* architecture [65].

Different inception module approaches have been implemented and the three most famous models are: *Inception-v3* [66], *Inception-ResNet-v2* [64], and *Xception* [13]. *Inception-v3* explores different filter sizes and combinations to create a purely inception module-based network. *Inception-ResNet-v2*, as the name indicates, incorporates residual blocks into its architecture. Consequently, this is the wider and deeper CNN architecture used in this research. Finally, the *Xception* network is a refinement of the *Inception-ResNet* models where, instead of residual blocks being forwarded into the network architecture, individual and convolutional filters with small transformations act as residual blocks. Nevertheless, both *ResNets* and inception models require a large amount of annotated training data [3], which is very scarce in most fine-grained categorization problems.

2.2.3 Data Augmentation and Fine-Tuning

Fine-grained categorization tasks suffer from the lack of training data, and therefore data augmentation techniques have been intensively used to train CNNs [9, 12, 24, 25, 30, 46, 68]. However, most approaches are limited to downsampling methods, resizing the original image or randomly extracted samples to fit into the first layer of the network. Some of the reviewed methods do not consider the aspect ratio of the original image and squeeze non-square samples to small square areas (generally 224x224 or 299x299 pixels). Furthermore, downsizing an image usually leads to information loss. To avoid losing information and to keep the original aspect ratio, different data augmentation approaches have been implemented. For example, extracting samples by cropping representative square areas from the center of the image has yielded good results when used to train CNNs [19, 24, 30, 46, 68, 69].

The most commonly used data augmentation method for training CNNs with insufficient data is random cropping [11, 69], which collects small representative areas randomly, aiming to select pieces of the object of interest. Extracting samples of different sizes from an image can also be used for data augmentation. The extracted samples can help improve the capability of CNNs to classify images in various scales correctly. A commonly used strategy is to feed these sampled areas to similar CNNs with different input layers [9, 12, 46] or with different depths [24, 68, 82].

Data augmentation has been further improved by two approaches: part-based patch extraction and segmentation. The part-based extraction of representative samples generally relies on manually annotated parts [8, 25, 44, 71, 80]. Consequently, most part-based extraction techniques are limited to part annotated datasets such as the CUB-200 [73], a dataset of 200 bird species and their annotated parts. As proposed by [4, 34, 32, 33, 59, 60], segmentation methods can assist in preparing images to train CNNs. These papers use segmentation to define the object's area, which is resized to fit in the first layer of the CNN. Considering that it is extremely difficult to define the parts of plants in natural images, patch extraction based on segmentation would be a more suitable approach for fine-grained categorization of plant species.

Another approach often used for training CNNs with insufficient data is *transfer learning* and *fine-tuning*. Transfer learning uses pre-trained CNN models with parameters learned through an exhaustive training process using millions of images [14]. The learned parameters are then transferred to a new CNN network and used to classify a specific dataset. Fine-tuning also uses pre-trained CNNs, but previously learned parameters are used as initial weights for a new training process. Hence, pre-trained CNNs are fine-tuned over the target dataset, starting their training process with previously learned parameters rather than random values. This approach has been widely used in fine-grained categorization problems that lack annotated training data.

2.2.4 Multi-Scale Approaches

Focusing on multi-scale approaches that handle the analysis of objects in natural images, I survey previous work and organize them as per their implemented method to address the scale issue.

Human-in-the-loop

The identification of plant species through flower image categorization is the objective of Cui *et al.* [16]. The multi-scale issue is addressed by normalizing all the extracted features to eliminate scale differences and correctly compute the distances in the designed feature space. However, this approach is not enough to handle the scale problem. For that reason, they implement a visual analysis (human-in-the-loop) with a botanist reviewing the final predictions. In this way, they re-integrate the incorrectly classified images into the dataset after this laboring classification.

Another approach using human-in-the-loop is implemented by Wah *et al.* [71] and is designed for the fine-grained categorization of birds. Their visual recognition system is composed of a

machine and a human user, who provides additional information by clicking on the object parts and answering binary questions. Using the CUB-200 dataset, Wah *et al.* [71] tackle the bird classification problem by analyzing specific areas of the image with the assistance of a user, who can easily indicate the bird parts (head, beak, body, wing, and tail) independent of the image scale.

Multi-scale fusion

Back to plant species categorization, Hu *et al.* [24] propose a multi-scale fusion CNN designed for leaf recognition. Using the MK Leaf [40] and the LeafSnap Plant Leaf [37] datasets, a custom CNN is trained by slowly infusing multiple resolution images with the list of bilinear interpolation operations used to sample them. In this way, downsized images are progressively fed to the CNN, concatenating extracted features at each level of the model to perform a multi-scale analysis. Nevertheless, their method is designed to work with leaf images taken in controlled backgrounds, limiting its application.

Implementing a classic approach called pyramid representation, Yoo *et al.* [77] create a pyramid multi-scale representation of the image to be analyzed by a pre-trained CNN. This analysis extracts dense activation vectors that are normalized and averaged by a pooling layer for the final classification. By creating pyramid representations of the analyzed images, this framework is able to perform a multi-scale fusion of features and outperform previously proposed methods on different datasets, including the Oxford 102 Flowers [49] and the MIT67 [74] for general indoor scenes.

Part-based image representation

In more recent work, Zhang *et al.* [80] present a supervised fine-grained categorization of bird species with part-based image representation. Basically, instead of collecting central samples, they propose to generate multi-scale part samples from random object parts, select the most useful ones, and use them to compute a global image representation for categorization. To select useful parts of the object (random cropping images that have pieces of the examined birds on them), they clustered all the part samples and explored useful information by computing an importance score that indicates how vital each cluster is for this fine-grained categorization task. Selecting samples from the most important clusters, they created a multi-scale CNN model for the categorization of bird species. However, during the random part selection process, samples of any sizes are considered to be rescaled and fit the first layer of the CNN. Therefore, these clusters also contain samples that do not respect the birds' aspect ratio, which probably impacts the model's performance.

Branson *et al.* [8] also propose a bird species categorization using pose normalized CNNs, and a graph-based clustering algorithm is used to learn a compact pose normalization space. In this case, cropped images of the bird's head, body, and the entire image pass through individual CNN streams. In both approaches [8, 80], the multi-scale issue is addressed by the extraction of random cropped images at various sizes, enabling CNNs to learn multi-scale invariant features randomly.

Nevertheless, they simply resize the cropped areas to small samples, changing the birds' aspect ratio. Paying attention to this detail, Liu *et al.* [44] present a similar multi-scale approach with the addition of "attention networks". These auxiliary models are independent CNNs implemented to identify sample areas at two different scale levels, combining the extracted multi-scale features for classification. As a result, this approach focuses on three different scales to extract and classify the bird's head, its body, and the entire scene, outperforming previously described methods in the fine-grained categorization of birds. However, their method relies on annotated object parts to construct the match between parts and classes, which makes it challenging to apply it to the categorization of plant species. To the best of my knowledge, there is no available dataset with plant images and their respective annotated parts (leaf, flower, fruit, etc.). Therefore, alternative methods to extract representative samples for fine-grained categorization of plants have to be designed.

Different feature representations

Multiple computer vision techniques have been proposed to solve the multi-scale issue, and some of these unconventional ideas can be adapted to the fine-grained categorization of plant species. As an example, Buyssens *et al.* [9] present a multi-scale CNN for the classification of cell images inspired by the different retina sizes of the human visual system. In their approach, each cell image is rescaled n times to fit the "retina" (or input layers) of the CNNs, and the same model is trained four different times with downsized samples of the input images. In this way, the multi-scale problem in the analysis of human cell images is addressed on a limited scale range.

Yuan *et al.* [78] also present a different feature representation. They develop a multi-scale and multi-depth CNN using a custom architecture designed for the remote sensing imagery pan-sharpening problem. The multi-scale capability of their model is designed by creating multiple convolutional filters inside each residual block. These filters have three different sizes and aim to extract multi-scale discriminative features. To a certain extent, this customized implementation is a simple version of the inception modules [65], which have mini-models inside of a bigger model. In conclusion, Yuan *et al.* [78] point out the importance of the multi-scale feature extraction implementing different sized filters for the pan-sharpening problem.

Segmentation-based approaches

Using segmentation approaches, Krause *et al.* [34] present an interesting idea for the fine-grained categorization of birds. They use annotated bounding boxes for training, but part annotations are not required during the classification process. Instead, they generate part samples using segmentation and alignment methods and combine them to represent the entire bird. Nevertheless, the use of annotated bounding boxes for training limits its application to datasets such as the CUB-200 [73] and the Cars-196 [35]. Nevertheless, the idea of segmenting the object to extract representative samples can be adapted for fine-grained categorization of plants.

With similar segmentation approaches, Angelova and Zhu [4] present a systematic object detection for fine-grained categorization of flowers. Their method first detects low-level regions that could potentially belong to the object of interest and then performs a full-object segmentation within those regions. They also zoom-in on the object, center it, and normalize its size to a single scale discounting the effects of the background. To understand the benefits of the segmentation step for fine-grained categorization tasks, Angelova and Zhu compare their approach with a baseline model. The baseline model does not use segmentation and is outperformed by their model in all tested datasets. Still, segmentation may be imperfect for most of the plant examples, and a robust approach should not depend entirely on this process.

2.2.5 Useful Insights

For plants in natural images, it is arduous to correctly identify all the details of a plant, distinguishing it from a complex background and other plants that may be in the same image. Approaches detailed in this chapter do not provide a final solution for the plant categorization problem when using natural images. Promising methods use part-based approaches to classify specific areas of the object of interest, but they rely mainly on part-annotated datasets that are not available for plants. The use of a human-in-the-loop (as a botanist) may also present impressive results but makes the plant categorization system less automated. Different feature representations and multi-scale fusion approaches generally seek the implementation of a customized CNN architecture with no pre-trained weights available, making these models harder to train over fine-grained categorization datasets.

As an insight from this survey, a viable contribution to the plant categorization problem would be a guiding system indicating where are the most representative areas in the image for the classification task. The plant categorization problem can also take advantage of the simultaneous analysis of different plant organs, combining their classification results to predict the plant species. In this dissertation, I explore existing segmentation approaches to create a plant localization system and carefully extract representative samples of plants and flowers. This process allows the classification methods to focus on the most important areas of the image individually. Furthermore, new strategies on how to use pre-trained CNNs for classifying unseen images at various scales could also present a valid contribution. In general, methods described in this chapter evaluate their approaches by classifying images only once during their classification processes. To provide a multi-scale analysis of a new image, I implement a novel classification process that explores the most representative areas of the query image and classifies them using CNN models fine-tuned to be more robust to scale variance. In this way, a single plant in a natural image can be analyzed from different perspectives, making the categorization system (*WTPlant*) more robust to variations in plant appearance at different scales.

CHAPTER 3
PLANT LOCALIZATION IN NATURAL IMAGES

In this chapter, I explore scene parsing approaches to locate different plant organs in natural images. By locating multiple plant parts, I address the first research question of this dissertation and identify the most representative areas for the fine-grained categorization of plant species. With the recent success of the CNNs, scene parsing approaches [43, 79, 81] are achieving excellent results when implementing convolutional blocks in their models. In particular, Zhou *et al.* [81] stack multiple convolutional blocks to create a new CNN model. In this way, they developed a cascade segmentation approach for the scene parsing problem (henceforth referred to as MIT Scene Parsing). This scene parsing approach segments a natural image into common semantic categories, including plants and flowers. Due to the highly accurate results reported on the segmentation of plants, this CNN model is the scene parsing method of choice for the *WTPlant* system.

3.1 Scene Parsing and Plant Segmentation

The MIT Scene Parsing, designed by Zhou *et al.* [81], implements a three-level CNN as cascade segmentation modules to parse a natural image into three main streams (stuff, object, and object parts). They trained this CNN model using ADE20K dataset [81] to segment 150 everyday objects, detecting general stuff (sky, road, building, etc.), objects (plant, car, people, etc.), and object parts (flowers, car wheels, people's heads and torso, etc.). Figure 3.1 shows this parsing model as being composed of three stacked convolutional blocks. In this CNN architecture, the first block is called Stuff Stream, for the background. The second convolutional block is the Object Stream, segmenting the foreground objects and adding them to the background ones. Finally, the combination of background and foreground generates the entire segmented scene. The third block, called Part Stream, is an optional one since not all objects have their parts annotated in the training dataset. When they do, this block takes the previously segmented foreground objects and adds the segmentation of their parts.

3.1.1 Pre-Segmentation of Flowers

Parsing a natural image brings the challenge of detecting small object parts, such as tiny flowers in a plant. The MIT Scene Parsing does not always capture them, and aggregating pre-segmentation approaches to assist in the detection of foreground objects can improve this process. Hence, I incorporate two pre-segmentation methods into the *WTPlant* system to assist the scene parsing on the detection of flowers. Called the Watershed Transform [56] and the GrabCut [57] algorithms, both methods separate the background from the foreground before the scene parsing and help on separate flowers.

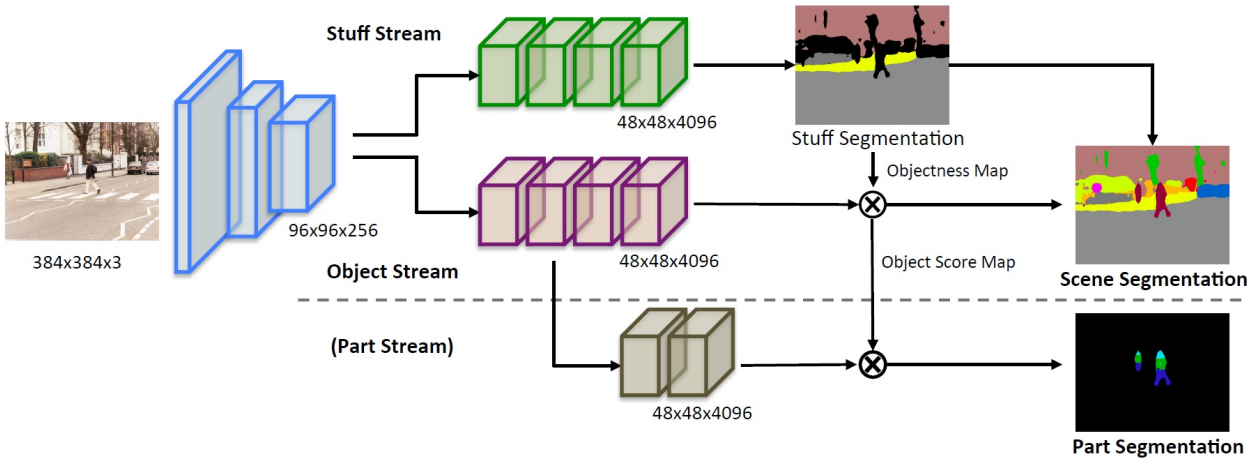


Figure 3.1: Cascade segmentation module of the MIT Scene Parsing CNN architecture [81].

Watershed Transform Algorithm

Meyer and Beucher [45] first introduced the Watershed Transform algorithm in the 1990s. This transformation is a morphological gradient-based segmentation process defined at a grayscale level. It segments the image as a topographic surface, by virtually flooding it from its minima while preventing the merging of the waters coming from different areas. For the flooding process, the gradient map of the image is considered as a relief map in which different gradient values correspond to different heights. Like this, the Watershed Transform algorithm partitions the image into watershed lines that separate the background from the foreground.

GrabCut Algorithm

Rother *et al.* [57] designed the GrabCut algorithm. This algorithm implements graphing representation to describe the boundaries of objects. The graph feeds an energy function that produces a proper segmentation when minimized. To perform the minimization, they built a graph where nodes represent pixels in the image and edges represent the difference in pixel color from one node to another. Using the Min-Cut/Max-Flow algorithm (which is a graph cut technique [7]), the resulted graph represents the segmented areas of the image and divides them into the foreground and background. As an example, Figure 3.2 presents a natural image of a *Mimosa pudica* plant and shows how these pre-segmentation methods assist the scene parsing process on detecting flowers.

To better understand the benefits of using pre-segmentation methods before the scene parsing, Figure 3.3 shows how the segmentation is performed using a natural image of the *Tabebuia berteroi* plant species. In this Figure, (a) presents the scene parsing results without any pre-segmentation methods applied. The MIT Scene Parsing successfully detects the plant and delineates the plant area (green) and flower area (red). These areas are called Regions of Interest (RoIs), and they guide the *WTPlant* system during the plant localization process. However, the scene parsing fails

(a) (b) (c)

Figure 3.2: (a) Example of a natural image of a *Mimosa pudica* and the results of pre-segmentation by (b) Watershed and (c) GrabCut.

to detect the flower in this image, identifying it as part of the plant. However, (b) shows that a pre-segmentation method (in this case, the Watershed Transformation) can assist the scene parsing by emphasizing the presence of flowers not detected previously. As a result, (c) presents a combination of the initial scene parsing and the Watershed pre-segmentation process locating plant and flower areas. If more than one RoI is detected, the largest areas are chosen to represent the plant and flower in the image. Furthermore, both RoI (plant and flower) have to be connected to guarantee that the largest flower belongs to the detected plant. If any RoI is collected, meaning the potential presence of plants or flowers in the image, the RoI is assumed to contain the most representative information of the plant and is further processed to predict the plant species. If no RoI is identified during the segmentation process, the image is considered as "No Plant Image".

(a) (b) (c)

Figure 3.3: Example (*Tabebuia berteroi*) of plant (green) and flower (red) segmentation. (a) Scene parsing without pre-segmentation, (b) scene parsing using the Watershed Transformation, and (c) combination of the largest plant and flower (using pre-segmentation approaches) areas.

3.2 Bounding Boxes of Segmented Areas

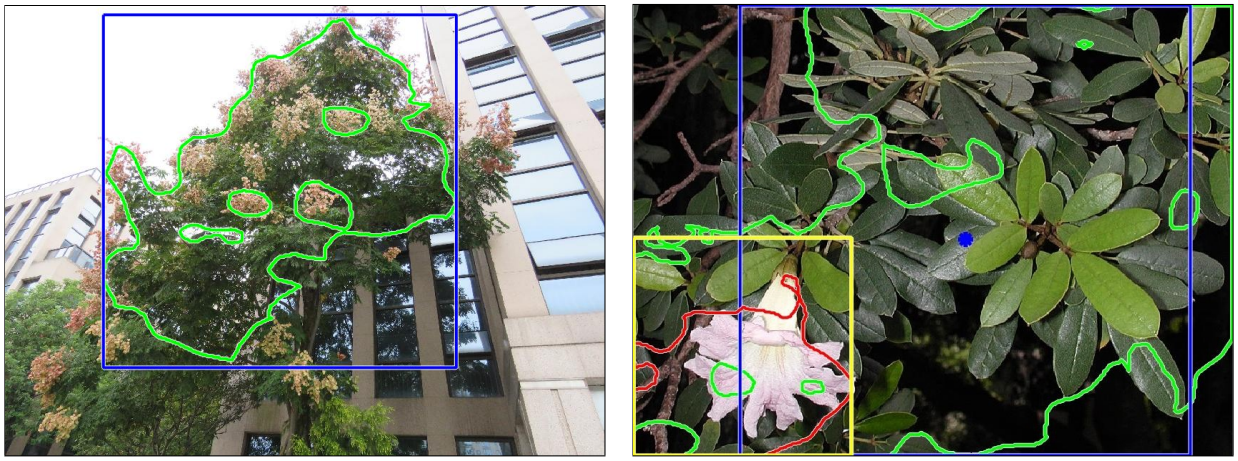
Defining bounding boxes is a necessary step implemented over the segmented areas so they can be inputted into the classification models. To make them fit, I rescale these bounding boxes to the size of the first layer of the models. CNN input layers generally receive square-shaped patches, which leads to the creating of square bounding boxes. Some of the reviewed approaches [5, 63, 70] suggest that simply downsizing the entire image is a good practice. But a drastic downscale of an image inevitably results in the loss of valuable information. Therefore, rather than using the input image as a whole, I implement a preparation method to collect the most representative samples out of the identified RoIs (plant and flower).

This preparation method starts by defining square bounding boxes to cover the RoIs, based on their minimum and maximum x and y coordinate values. Using the difference between these coordinates, I calculate the width and height of the plant and flower RoIs. The larger value between the width and height of each RoI defines the size of the bounding box. To ensure that all bounding boxes are inside the image, the size of these boxes has to be less than or equal to the minimum between the width and height of the input image. An essential aspect of this method is that all the bounding boxes are square-shaped. In this way, extracted samples will keep the aspect ratio of the image, without stretching or squeezing the plant sample when resized to fit the first layers of the classification CNNs.

Figure 3.4 shows plants in natural images, their RoIs segmented by the scene parsing stage, and the bounding boxes delimiting the most representative areas of the image for the plant categorization problem. More specifically, Figure 3.4 (b) shows the positioning of bounding boxes for plant and flower. In this example, plant width is greater than the height of the image, so the bounding box is limited to the size of the image and guided by the centroid (blue dot).

3.3 Initial Experiments

The initial experiments aim to compare the accuracy of a CNN model when trained using different approaches, with (Bounding Box) and without (Resize and Central Crop) plant localization. Using the *ResNet* models, I train CNNs with different depths by simply resizing the images (approach implemented by [5, 63, 70]), using central cropped samples (implemented by [30, 46, 24, 68]), and samples extracted using the bounding boxes. The objective of these initial experiments is to verify if the selected square bounding boxes are the most representative areas of the image for the plant categorization problem. If they are, the CNNs should present more accurate results when trained using these areas. Experiments described in this dissertation used *MATLAB R2015b*, *Python 3.6*, and *Keras 2.2.4 API*. The testbed is *Ubuntu 16.04* operating system with a *NVIDIA GeForce GTX 1080* GPU used to train the CNNs.



(a) (b)

Figure 3.4: Example of Bounding Square-Shaped Boxes. (a) *Koelreuteria formosana* plant selected area (blue), and (b) *Tabebuia berteroi* plant (blue) and flower (yellow) selected areas.

3.3.1 UHManoa100 Dataset

Focusing on plants present in the University of Hawai'i at Mānoa (UHM) campus, I collect a dataset named UHManoa100 with a total of 4,778 natural images of 100 plant species. The Botany Department of the UHM kindly shared most of these annotated images, and species with fewer images have more added after scraping new ones from the *bing.com* search engine website. Appendix A presents the complete list of the plant species selected for this dataset.

For each image in the UHManoa100 dataset, the annotated plant species indicates the dominant plant present in the image. Different plants may appear in the background or even in front of the dominant plant, but each annotated plant species covers the largest area of the image. Another important characteristic of this dataset is that images have different resolutions (ranging from 300x300 to 6000x4000 pixels) with varying orientations and locations of the plants. As shown in Figure 3.5, these images contain plants at various scales ranging from the leaf or flower to the entire bush or tree. Using the UHManoa100, I create a balanced training set by selecting 4,500 of these images, 45 images per each of the 100 plant species. For each plant species, a test set of visually challenging images for this categorization problem is put aside for performance evaluation. Testing images also have plants at various scales and showing multiple organs (leaf, flower, fruit, bush, and tree). They comprise a testing set of 278 images unseen by trained models.

3.3.2 Metrics and Initial Results

To better understand the initial experiments, Figure 3.6 shows how the three preparation approaches extract representative samples from images of the UHManoa100 dataset. I extract different samples by resizing the entire image (Resized) to fit the first layer of the classification CNN, by

Cascabela thevetia

Cyperus papyrus

Figure 3.5: Plant images from the UHManoa100 dataset.

cropping and then resizing the central area of the image (Central Crop), and by extracting guided samples around the area determined by the plant localization (Bounding Box).

Initial experiments start by extracting samples from the training images of the UHManoa100 dataset and randomly dividing them into 80% for training and 20% for validation. I train all the *ResNet* models selected for these experiments for 100 epochs with hyperparameters set as suggested by He *et al.* [20]. I use the backpropagation algorithm to propagate the error backward throughout the CNN and update its parameter values (weights and biases). This process is performed for each training sample while using the corresponding validation data to calculate the training accuracy at each epoch. During the training process, *ResNet* models are not overfitting after 100 epochs. I observe this behavior by monitoring performance on the validation set and stop the training at 100 epochs because I am limited by computation. The final trained model is the one with the smallest validation error after completing the training process.

I evaluate the trained CNNs using the testing set of images unseen by the trained models, containing at least one sample of each plant species. Respective samples (Resized, Random Crop,

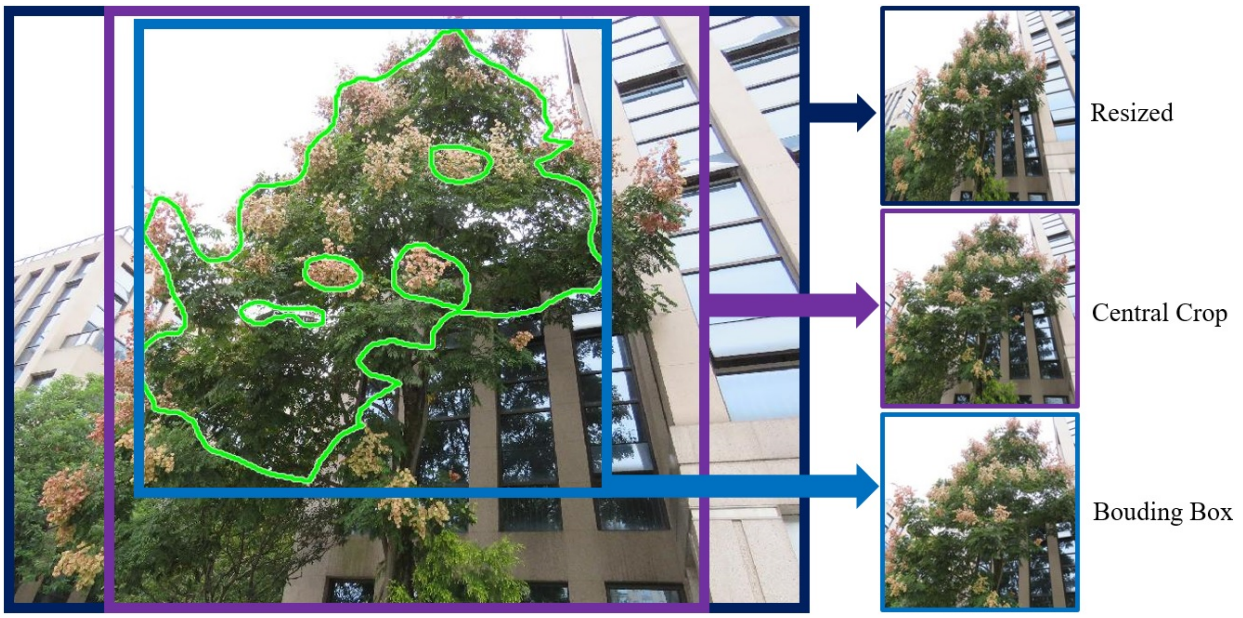


Figure 3.6: Preparation approaches with and without plant (*Koelreuteria formosana*) localization.

and Bounding Box) are extracted from the test images to perform the evaluation. As a metric, I use the prediction accuracy, e.g., the percentage of images correctly categorized in the testing set. And I consider the image correctly categorized when the Top-1 prediction matches the annotated species of the plant. Initial experiments categorize only plants, the analysis of flowers is added later. Table 3.1 presents the accuracy results of *ResNet* models with 18, 34, and 50 layers. In the initial experiments, I train the CNN models using three different areas of the images to compare the representativeness of the plants in the extracted samples. After 100 training epochs, *ResNet18* is the CNN that learns more discriminative features when trained using the bounding boxes. Deeper models generally require more epochs to be fully trained, especially with no pre-trained knowledge. The lack of training data also impacts the generalization capability of these models.

For the three sample collecting methods (Resized, Central Crop, and Bonding Boxes), initial results show the bounding boxes as the best approach to extract samples for the training of plant classification models. The central crop approach presents similar results, but this method would likely collect irrelevant information for the plant categorization task when extracting samples from the UHManoa100 images. Furthermore, this comparison shows that the most valuable information is not necessarily in the central portion of the images. Therefore, a guiding process to the specific location of the plant can improve the extraction of better training samples. The resized preparation method is commonly used for the training of numerous classification models, but it does not take into consideration the aspect ratio of the plants. In this case, it is outperformed by methods that better handle training images varying in size, orientation, and resolution, such as the images from the UHManoa100.

Table 3.1: Initial accuracy percentage of correctly categorized UHManoa100 testing images.

CNN Model	Resized	Central Crop	Bounding Boxes
ResNet18	39.21%	44.24%	**45.32%**
ResNet34	40.29%	42.81%	43.17%
ResNet50	28.42%	39.21%	40.65%

3.4 Pseudocode I

The following pseudocode (Algorithm 1) details the plant and flower localization process designed to identify the most representative areas in an image for the plant species categorization problem. Receiving a query image (I), the "PlantFlowerLocalization" algorithm uses a scene parsing approach to create the plant and flower segmented areas (I_{Plant} and I_{Flower}). It also calculates the respective bounding boxes ($BBPlant$ and $BBFlower$) defining the most representative areas for this categorization task.

3.5 Observations and Discussions

In this chapter, I extract the most representative areas of an image for the plant species categorization by defining bounding boxes around the detected plant organs. This approach has shown promising results, and it is the first contribution of this dissertation. The implemented method locates plants and their flowers using a scene parsing approach to guide the delimitation of the most representative areas in the image for the classification of plants. This method also qualitatively improves the segmentation of flowers (Figure 3.3) by implementing two pre-segmentation approaches.

The MIT Scene Parsing is further tested on detecting the presence of plants and flowers over all the 4,778 images of the UHManoa100 dataset. As a result, this scene parsing CNN can recognize the presence of plants in 99.24% of training and testing images. I credit this highly accurate result to Zhou *et al.* [81] and their work on the MIT Scene Parsing. It guides the extraction of more representative samples on almost all of the training images. For those images in the training set that do not have their plants detected, I extract the central crop areas and use them as representative samples. Figure 3.7 presents some of these training images that are initially classified as "No Plant" but are integrated into the training set using central cropping. They are extreme close-up shots that may not have the classical plant characteristics (green leaves, bushes, flowers, etc.) and, as a consequence, MIT Scene Parsing does not detect the presence of plants.

Initial experiments comparing the resized, central crop, and bounding box approaches have shown interesting results. As presented in Table 3.1, CNNs trained with bounding boxes around localized plant areas performed better when compared to the other approaches. This is probably related to the fact that some plants in the UHManoa100 dataset are not necessarily centralized

Acalypha wilkesiana

Artocarpus altilis

Eucalyptus deglupta

Artocarpus altilis

Figure 3.7: Natural images classified as "No Plant" by the plant localization process.

in the image. Therefore, a central crop may be excluding important discriminative information in its process. On the other hand, the resized method uses the entire image to extract the samples and may contain irrelevant information to the plant categorization task. Besides that, samples produced by this approach generally do not respect the aspect ratio of the plant and change its morphological characteristics. As a result, accuracy values reported in Table 3.1 support the idea that *ResNet* models' training process can be improved by feeding only the most representative area of the image instead of a general central crop or even the entire resized image. Furthermore, I can exploit these representative areas defined by the bounding boxes in a way that more samples are extracted to accommodate the lack of data during the training process of deep classification models such as the *ResNets*.

Algorithm 1: PLANTFLOWERLOCALIZATION (I)

Input: Image I
Output: I_{Plant} , $BBPlant$, I_{Flower} , $BBFlower$

/ Scene Parsing with Two Pre-Segmentation Methods for Flowers */*
$I_{Plant}, I_{Flower1} \leftarrow$ LargestRoI (SceneParsing (I)) ;
$I_{Watershed} \leftarrow$ WatershedAlgorithm (I) ;
$I_{GrabCut} \leftarrow$ GrabCutAlgorithm (I) ;
$I_{Flower2}$, $I_{Flower3} \leftarrow$ LargestRoI (SceneParsing ($I_{Watershed}$, $I_{GrabCut}$)) ;
$I_{Flower} \leftarrow$ LargestRoI ($I_{Flower1}$, $I_{Flower2}$, $I_{Flower3}$) ;

/ Define Bounding Box if Region of Interest is Detected */*
if $I_{Plant} \neq 0$ **then**
  min_x , $min_y \leftarrow$ MinCoordinates (I_{Plant}) ;
  max_x , $max_y \leftarrow$ MaxCoordinates (I_{Plant}) ;
  $Width \leftarrow max_x - min_x$;
  $Height \leftarrow max_y - min_y$;
  if $Width > I_{height}$ || $Height > I_{width}$ **then**
    $BB_{size} \leftarrow$ min ($Width$, $Height$) ;
    $center_x, center_y \leftarrow$ CenterOfMass (I_{Plant}) ;
  else
    $BB_{size} \leftarrow$ max ($Width$, $Height$) ;
    $center_x, center_y \leftarrow$ GeometricCenter (I_{Plant}) ;
  $BBPlant_{TopLeft} \leftarrow$ [($center_x - BB_{size}/2$) , ($center_y - BB_{size}/2$)] ;
  $BBPlant_{BottomRight} \leftarrow$ [($center_x + BB_{size}/2$) , ($center_y + BB_{size}/2$)] ;
  return (I_{Plant} , $BBPlant$)
if $I_{Flower} \neq 0$ **then**
  min_x , $min_y \leftarrow$ MinCoordinates (I_{Flower}) ;
  max_x , $max_y \leftarrow$ MaxCoordinates (I_{Flower}) ;
  $Width \leftarrow max_x - min_x$;
  $Height \leftarrow max_y - min_y$;
  if $Width > I_{height}$ || $Height > I_{width}$ **then**
    $BB_{size} \leftarrow$ min ($Width$, $Height$) ;
    $center_x, center_y \leftarrow$ CenterOfMass (I_{Flower}) ;
  else
    $BB_{size} \leftarrow$ max ($Width$, $Height$) ;
    $center_x, center_y \leftarrow$ GeometricCenter (I_{Flower}) ;
  $BBFlower_{TopLeft} \leftarrow$ [($center_x - BB_{size}/2$) , ($center_y - BB_{size}/2$)] ;
  $BBFlower_{BottomRight} \leftarrow$ [($center_x + BB_{size}/2$) , ($center_y + BB_{size}/2$)] ;
  return (I_{Flower} , $BBFlower$)
if $I_{Plant} = 0$ *and* $I_{Flower} = 0$ **then**
  return (*No Plant Detected*)

CHAPTER 4
MULTI-SCALE PLANT CATEGORIZATION SYSTEM

In this chapter, I present a novel plant categorization system named *WTPlant* (*What's That Plant?*). This system implements a multi-scale preprocessing method for the classification of plants in natural images. This method focuses on extracting multiple guided samples at different scales from plant and flower areas (bounding boxes) to train the classification models. These samples are square-shaped to maintain the aspect ratio of the plant while fitting the first layer of the CNNs. More precisely, *WTPlant* searches for the most representative squared areas over the detected RoIs, extracts guided multi-scale samples from these areas, and trains its classification models focusing on different organs of the plant.

Using the bounding boxes (Section 3.2) and the guided multi-scale preprocessing method, I implement the *WTPlant* as a collection of CNNs (CNN-based) working together to solve the plant species categorization problem using natural images. I designed it to handle problems such as: *i)* If there is a plant in the picture, locate the most representative areas of the image for this categorization task; *ii)* The need for classification models more robust to variations in the scale of the plant. In particular, the preprocessing method of this system works as guided multi-scale data augmentation, making CNN models more robust to plant variations when trained with extracted samples. Experiments performed in this chapter show an improvement in the CNNs accuracy when using guided multi-scale samples for their training process.

The *WTPlant* system brings together different CNNs to meet the challenges of identifying plants in natural images. These challenges consist of localizing the plant in a complex natural scene, dealing with the multi-scale problem, and implementing a suitable CNN model deep enough to extract discriminative features among similar plant species. *WTPlant* addresses these issues by using stacked convolutional blocks for the localization of plant and flower areas, a preprocessing method to perform multi-scale analyses, and Residual Blocks [20] and Inception Modules [65] to extract deeper and more discriminative features.

In Section 4.1, I describe the framework and pipelines of this new plant categorization system, emphasizing its modular capability. A guided multi-scale data augmentation presented in Section 4.2 trains the CNN models of this system using samples extracted at various scales. Section 4.3 details how the use of multi-scale samples during the categorization of each test image improves the classification process of each pipeline. Experiments show that the guided multi-scale data augmentation and the new classification process improve the models' accuracy. I also investigate the use of different CNNs (implementing inception modules) in Section 4.4, and the importance of pre-trained weights during the fine-tuning of these models. Furthermore, in Section 4.5, a Graphical User Interface (GUI) shows each stage of the *WTPlant* system (localization, preprocessing, and classification) during the categorization of a new image.

4.1 Framework and Pipelines

The *WTPlant* system consists of separate pipelines and multiple stages of CNN components designed to extract deep discriminatory and scale-invariant features. Figure 4.1 presents the framework of this system and details the workflow during the categorization of the plant in an input image. This framework has two pipelines for the analysis of plants and flowers simultaneously, and they start with the segmentation and localization of the most representative areas for each classification problem. For flowers, two pre-segmentation algorithms assist the main scene parsing stage. After the scene parsing, the preprocessing method creates scale representative samples of the largest segmented areas to feed the classification models. Next, CNNs individually trained for each plant or flower categorization task classify the extracted samples. In the final stage, the prediction confidence analysis of each plant and flower sample helps predict the plant species. In summary, *WTPlant* uses a varied number of multi-scale samples extracted through the guidance of the scene parsing to classify them individually, combining their predictions over different plant organs at various scales. Thus, the analysis of the plant in a natural image can be performed over a wide range of scales, making this system more robust to the scale variance.

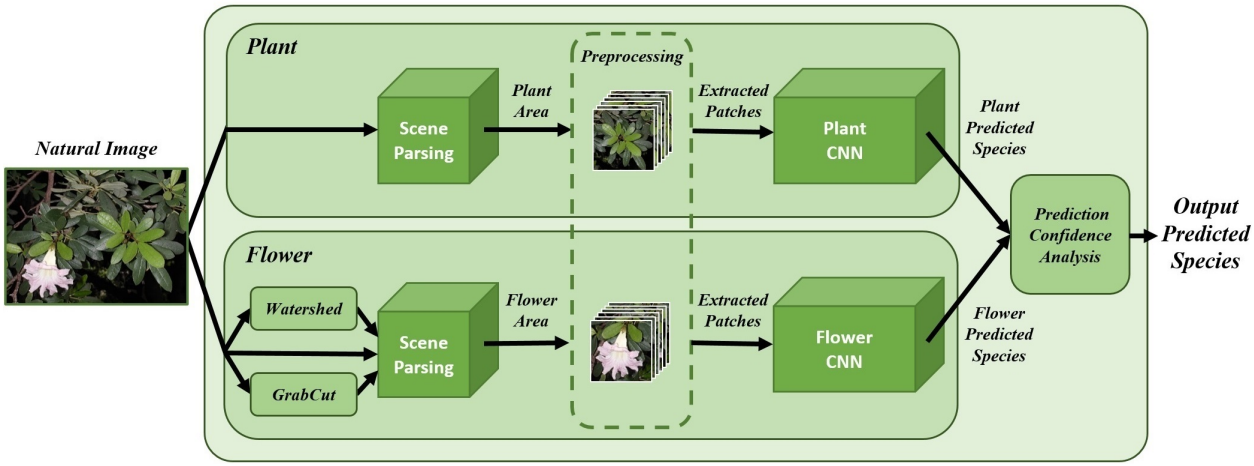


Figure 4.1: Overview of the *WTPlant* system.

In contrast to existing plant identification methods that use hand-designed features or simple CNN architectures, *WTPlant* implements a collection of CNNs to classify plants and flowers separately, and then combine their predictions to achieve a more accurate categorization of the plant species. By designing two classification pipelines, one for general plants (leaves, branches, bushes, and trees) and one specifically for flowers, *WTPlant* can handle natural images with plants, flowers, or both together. Other organs of the plant, such as fruit, bark, root, seedling, etc. can be similarly classified and integrated into the system by adding new pipelines. In this way, this framework can be expanded to a multi-objective fine-grained categorization problem and become the ultimate plant identification system.

4.1.1 Modularity

WTPlant framework incorporates auxiliary methods in a modular fashion, enabling the improvement of each stage of the system (localization, preprocessing, and classification) to be done independently. Moreover, each CNN can be easily upgraded (fine-tuned or retrained) and incorporated back into the system. Therefore, later versions of the *WTPlant* system can be more accurate by introducing new and more powerful CNNs. These new models can also be scaled up to classify a broader range of plant species, covering a much larger environment. Experiments described in this dissertation support the idea that a collection of CNNs carefully designed to analyze multiple plant organs simultaneously may overcome the limitations of commonly used methods and monolithic, non-modular DL approaches for the plant categorization problem.

4.2 Guided Multi-Scale Data Augmentation

As suggested by initial experiments (Section 3.3), bounding boxes delimiting the plants' detected regions are the most representative areas of the images for the plant categorization task. These areas can be further exploited by extracting samples at different scales, depending on the resolution of the image. So I exploit these areas implementing a new preprocessing method that changes the scale of the plant and flower by zooming into its minimum resolution (minimum number of pixels in the selected region without resizing it). The center of mass or centroid of each RoI is an excellent indicator of the plant/flower location and, excluding extreme cases, its coordinates are inside of the RoI. Using the coordinates of these centroids as their centers, I define the close-up areas for plant and flower by selecting the input size of the CNN (224x224 or 299x299 pixels) and extracting the most "zoomed-in" samples at a minimum resolution, called the close-up patches. In this case, patches are the extracted samples used to train the classification models.

4.2.1 Extracting Multi-Scale Representative Patches

The close-up areas define the first patches used to train plant and flower classification models of the *WTPlant* system. The extraction of patches at multiple scales starts by collecting the close-up area and keeps extracting larger areas until it reaches the respective bounding box. When used for training, this process is called the guided multi-scale data augmentation. In cases that the bounding boxes are smaller than the close-up area, I extend these boxes to the borders of the image, creating a space between the bounding box and the close-up patch. Typically, the bounding box is bigger than the close-up area (depending on the resolution of the image). In both cases, the desired number of patches divides the difference between the bounding box and the close-up area. More specifically, the guided multi-scale extraction process uses the top-left and bottom-right coordinates of the bounding box and the close-up area to calculate the increasing value between the multi-scale patches.

As an example, Figure 4.2 presents this process by showing the RoI of the plant (green), its bounding box (the largest square in blue), the centroid of the RoI (blue dot), the close-up area (the smallest box in blue), the multi-scale samples extracted (intermediate boxes in blue), and the coordinates (red dots) used for positioning the multi-scale patches to be extracted. In this example, an area of 224x224 pixels (size of the first layer of the *ResNet* models) around the centroid of the RoI defines the close-up area and directs the extraction of larger areas. I later resize these areas using nearest-neighbor interpolation to fit the first layer of the CNN, creating the multi-scale patches used to train the classification models.

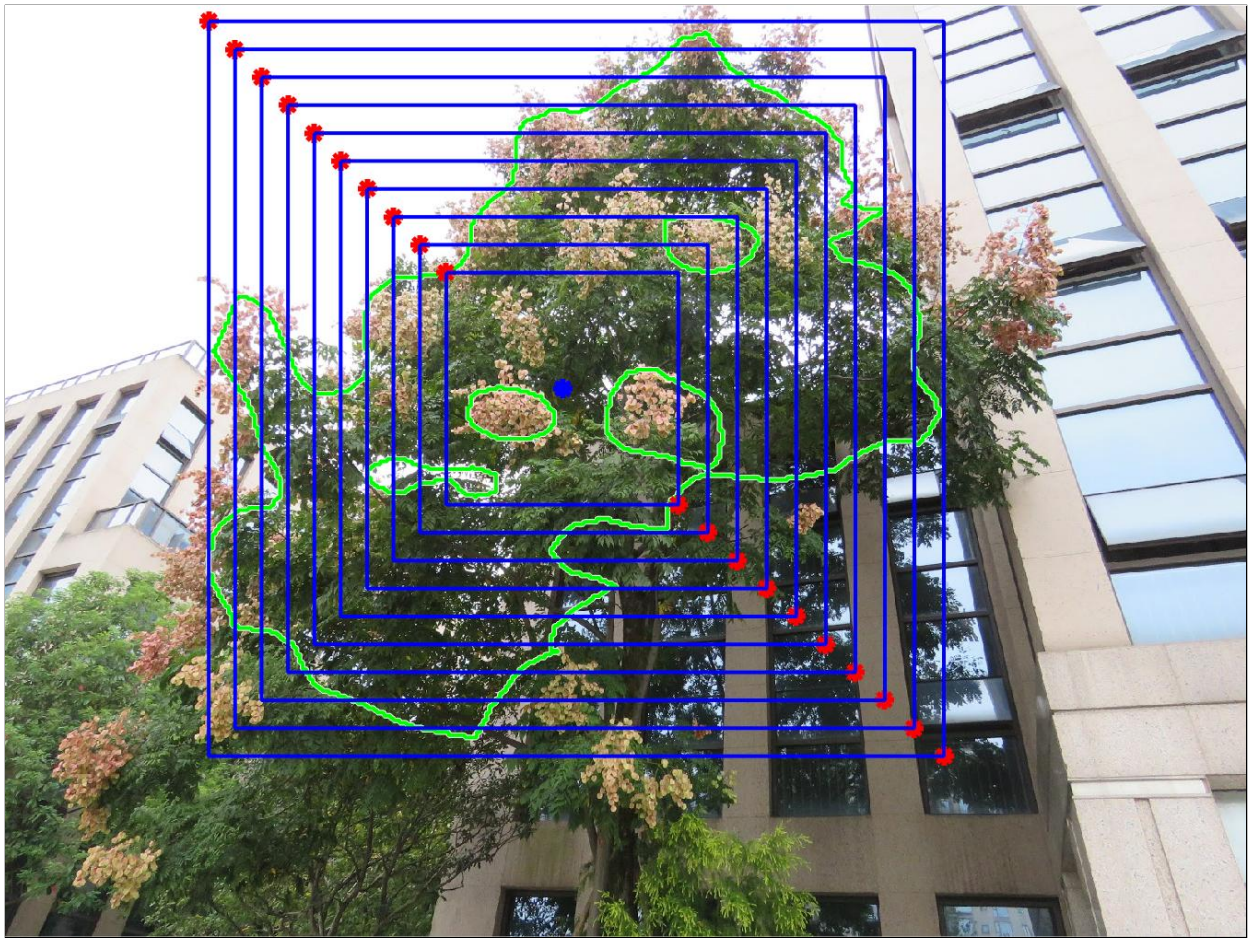


Figure 4.2: Example of guided multi-scale data augmentation process. The scene parsing localize the plant (*Koelreuteria formosana*) region (green). *WTPlant* uses the coordinates of the bounding box, the close-up area, and the patches in between (red dots), to collect multi-scale patches (blue).

After the preprocessing method, all extracted patches fit the first layer of the selected classification model without changing the plant aspect ratio. Thus, this method is a suitable data augmentation for natural images of plants, with a variable number of multi-scale patches to be extracted. As an example, Figure 4.3 presents ten extracted patches resized and ready to be fed

into the first layer of the classification CNN during the training process. With variable size and number of patches to be extracted, this guided multi-scale data augmentation is limited only by the resolution of the training images. Applying this method to train the classification CNNs of the *WTPlant* system, I create DL models that are more robust to scale variations of plants and flowers in natural images.

Figure 4.3: Patches from Figure 4.2 resized to fit the first layer of the *ResNet* models.

4.2.2 Pseudocode II

The following pseudocode (Algorithms 2 and 3) details the preprocessing method of the *WTPlant*. I designed it to make the system capable of categorizing plants and flowers at different scales. The returned set of multi-scale patches ($\mathcal{M}_{WTP}$) are extracted, resized according to the size of the first layer of the CNN, and used as augmented data during the training process.

4.2.3 Experiments

Experiments described in this Section verify the efficacy of the guided multi-scale data augmentation method when training CNN models for the plant categorization task. They used different datasets to compare the guided multi-scale data augmentation against a commonly used approach called the random crop. As suggested by its name, the random crop data augmentation approach chooses arbitrary samples from random regions of the training images. These random samples are extracted to create training patches, which have to be resized to fit the first layer of the classification models. During the resizing process, it is important to take into consideration that the input layer of the classification models is usually square-shaped, and randomly extracted patches have to fit that shape perfectly. Because of that, most implementations of the random crop not always respect the aspect ratio of the plants when extracting patches.

Algorithm 2: GuidedMSPatchExtraction (I)

Input: Image I
Output: A Set of Multi-scale Patches $\mathcal{M}_{WTP}$

/ Configurable Parameters */*
$n \leftarrow 10$; *// Number of Multi-Scale Patches*
$p \leftarrow 244$ or 299 ; *// Patch Size for ResNets or Inception Models*
/ Plant Localization */*
(I_{Plant} , $BBPlant$, I_{Flower} , $BBFlower$) $\leftarrow$ PlantFlowerLocalication (I) ;
/ Guided Multi-Scale Data Augmentation */*
$\mathcal{M}_{Plant} \leftarrow$ GuidedMSDataAugmentation (I_{Plant} , $BBPlant$, n , p) ;
$\mathcal{M}_{Flower} \leftarrow$ GuidedMSDataAugmentation (I_{Flower} , $BBFlower$, n , p) ;
$\mathcal{M}_{WTP} \leftarrow$ ($\mathcal{M}_{Plant}$, $\mathcal{M}_{Flower}$) ;
return ($\mathcal{M}_{WTP}$)

A single image can produce a large number of randomly extracted samples. For the following experiments, I randomly create ten training patches for each plant image. In this way, CNNs trained using random crop have ten times more training data than models trained with previous approaches (Section 3.3). The guided multi-scale approach also extracts ten patches from each plant image to provide a fair comparison between the data augmentation methods. The main difference is that, instead of being random areas, this new data augmentation approach extracts patches at various scales, zooming into the most representative areas of the image, guided by the centers of mass (centroids) of the segmented plant and flower areas.

Using the same *ResNet* models from initial experiments, I similarly train the CNNs by simply resizing the input images, using randomly cropped patches at the size of a quarter of the total area of the image, and using patches extracted by the new guided multi-scale data augmentation. The objective is to verify if patches extracted at different scales help the trained CNNs become more robust to scale variations and better recognize plants in natural images. Two datasets with annotated plants are used: the BJFU100 [63] from the Beijing Forestry University, and the UH-Manoa100 (Section 3.3.1) from the UHM, both with 100 different plant species. All images used to train and test the CNNs are natural images, presenting complex backgrounds, partial occlusions, shadows, varying illumination, and different objects in the same scene.

BJFU100 Dataset

Recently, a collection of annotated high-resolution images called BJFU100 is presented by Sun *et al.* [63]. The BJFU100 dataset has 100 images per plant species, totalizing 10,000 natural images of ornamental plants present on the campus of the Beijing Forestry University. Figure 2.3 shows examples of these images. To better understand this dataset, Figure 4.4 shows some of the images

Algorithm 3: GUIDEDMSDATAAUGMENTATION (I_{Mask} , BB , n , p)

Input: Mask and Patch Info I_{Mask} , BB, n , p
Output: Multi-scale Patches $\mathcal{M}$

/* Define Close-Up Patch */
$centroid_x, centroid_y \leftarrow$ CenterOfMass (I_{Mask}) ;
$CloseUp_{TopLeft} \leftarrow$ [($centroid_x - p/2$) , ($centroid_y - p/2$)] ;
$CloseUp_{BottomRight} \leftarrow$ [($centroid_x + p/2$) , ($centroid_y + p/2$)] ;
/* Define Increasing Ratio */
$Ratio_{TopLeft} \leftarrow$ [($BB_{TopLeft} - CloseUp_{TopLeft}$) / ($n - 1$)] ;
$Ratio_{BottomRight} \leftarrow$ [($BB_{BottomRight} - CloseUp_{BottomRight}$) / ($n - 1$)] ;
/* Extracting Multi-Scale Patches */
$\mathcal{M} \leftarrow \emptyset$;
for $i \leftarrow 0$ **to** $n - 1$ **do**
    $TopLeftCorner(i) \leftarrow CloseUp_{TopLeft} - i \times Ratio_{TopLeft}$;
    $BottomRightCorner(i) \leftarrow CloseUp_{BottomRight} + i \times Ratio_{BottomRight}$;
    $patch_{MultiScale} \leftarrow$ ExtractPatch ($TopLeftCorner(i)$, $BottomRightCorner(i)$) ;
    $patch \leftarrow$ Resize ($patch_{MultiScale}$) to $p \times p$;
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{ patch \}$;
return $\mathcal{M}$

from the same category in the dataset. It is noticeable that some of these images were taken from the same specimen, facilitating the classification task of this dataset.

In their experiments, Sun *et al.* downsized these images to collect training patches for customized residual networks (*ResNets*). In this dissertation, I perform a comparison between their results and accuracies achieved using the same CNN models but three different preprocessing methods. Experiments detailed here followed the same implementation of Sun *et al.* and split the dataset by 80% for training and 20% for testing. The random crop approach extracts the same number of patches used by the guided multi-scale data augmentation, and both resize their patches to 224x224 pixels (as suggested by He *et al.* [20]) to fit them into the first layer of the *ResNets*. Three *ResNet* models are trained for 100 epochs, and Table 4.1 presents the resulting prediction accuracy. These results support the hypothesis that a guided multi-scale data augmentation can assist in the training of CNN models. More specifically, this new method outperforms the random crop data augmentation and the commonly used resized approach. It is noticed that the guided multi-scale data augmentation takes advantage of the high-resolution images of the BJFU100 dataset to extract representative patches of the plants. Meantime, resized and random crop approaches extract samples from a single scale. Consequently, models trained with these data augmentation methods do not result in CNNs robust to scale variance on the plant appearance. When compared with results from Sun *et al.*, the multi-scale data augmentation also shows a significant improvement.

Figure 4.4: Example images of two plant species from the BJFU100 dataset.

Even improving the *ResNet* models, Sun *et al.* need to implement a suitable preprocessing method to better prepare their natural images for the plant categorization task.

For the guided multi-scale data augmentation, only the plant area is considered from the scene parsing since this dataset does not present specimens with flowers. To be a fair comparison between the methods, I use the guided multi-scale data augmentation only for the training process. The classification is performed over the extracted bounding boxes of the testing images. That is, CNNs classify only one patch per testing image for all approaches, and the largest squared central crop tests the Random Crop. Sun *et al.* also implemented a customized *ResNet* with 26 layers, which resulted in their best accuracy of 91.78%. However, it is still below the performance of the *ResNet18* trained using the guided multi-scale approach, which yielded the best accuracy of 96.85%.

Table 4.1: Percentage of accurately categorized BJFU100 test images.

CNN Model	Resized	Random Crop	Sun *et al.* [63]	Multi-Scale Data Aug.
ResNet18	74.33%	87.78%	89.27%	**96.85%**
ResNet34	71.38%	85.53%	88.28%	96.65%
ResNet50	53.73%	73.73%	86.15%	91.15%

For these experiments, the BJFU100 dataset provides a fair amount of high-quality annotated plant images. It is noticeable in Figure 4.4 that these images are well standardized with single size (3120x4208 pixels), all center-oriented, with small light variations, showing almost no occlusions, and most importantly, having similar scales between images of the same species. These aspects make the BJFU100 dataset relatively easy to classify, which explains the highly accurate performance presented in Table 4.1. For example, Figure 4.5 visualizes the normalized distribution of the guiding points – the center of mass or centroids of segmented plant area. This heatmap shows that BJFU100 plant images are mostly center-oriented, making it easy for data augmentation approaches to collect representative patches. Even though BJFU100 is a useful dataset for data augmentation experiments, it does not present a wide range of scale changes. As a result, the CNNs trained on the BJFU100 dataset may not have learned scale-invariant features because of the small intra-species scale variation in the training data. Therefore, a model that aims to categorize plants in natural images ranging from leaves to entire trees has to take into account the multi-scale issue and be tested on a dataset with a large scale variation in plant appearance.

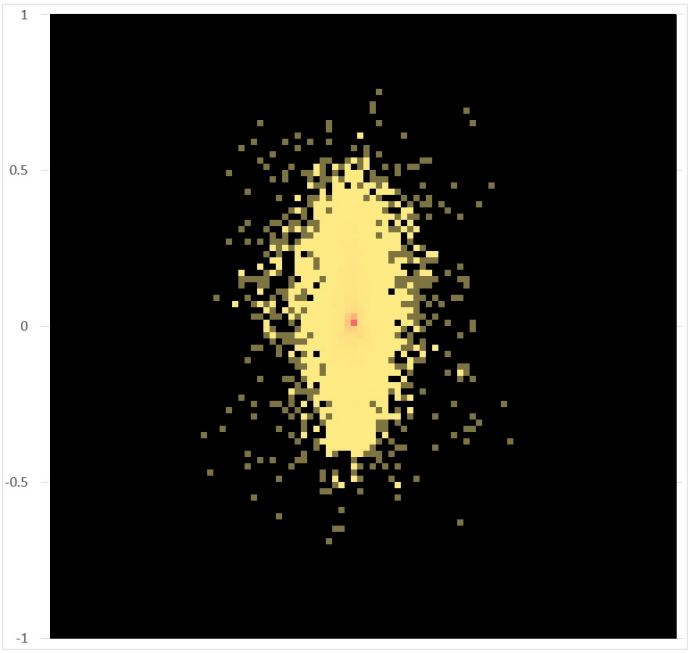


Figure 4.5: Normalized heatmap of 10,000 centroids from BJFU100 plant images, indicating that plants are located around the center in most images of this dataset.

UHManoa100 Dataset

I presented the UHManoa100 dataset in Section 3.3.1, and the following experiments using these plant images seek to compare the novel guided multi-scale data augmentation with commonly used methods in a more complex dataset. As presented in Figure 3.5, plants from the UHManoa100 dataset have images at different scales, varying their size and resolution. I collected images of this dataset from different sources and not from the same specimen, resulting in a wide variety of plant appearance. Consequently, accuracy results from UHManoa100 experiments cannot be directly compared with experiments using the BJFU100 dataset. In both cases, I implement the guided multi-scale data augmentation in an attempt to make the trained CNNs more robust to scale variations. For the UHManoa100, comparative results of prediction accuracy from this new data augmentation approach with other methods are presented in Table 4.2. As in previous experiments, the *ResNet18* is the CNN model that achieved the highest accuracy result after 100 training epochs. These results also show that the UHManoa100 dataset is difficult to categorize, and preprocessing methods that augment the data (Random Crop and Multi-Scale Data Aug.) help in the training of these classification models.

Table 4.2: Percentage of accurately categorized UHManoa100 test images.

CNN Model	**Resized**	**Random Crop**	**Multi-Scale Data Aug.**
ResNet18	39.21%	43.89%	**49.28%**
ResNet34	40.29%	44.60%	47.84%
ResNet50	28.42%	43.53%	47.48%

Experiments over the UHManoa100 show how this dataset is different from the BJFU100. More specifically, the locations of the plants in the images of the UHManoa100 dataset are scattered and not centralized as much as the images of the BJFU100 dataset. Figure 4.6 shows the normalized heatmap of the centroids from segmented plant areas of the UHManoa100 dataset. For this dataset, the preprocessing stage of the *WTPlant* system can localize the plants better than other approaches and successfully extract more representative patches for the training process of the CNNs. In this way, I exploit the most representative area of the images for the plant categorization task by implementing a guided multi-scale approach and creating a novel data augmentation process. This is the second contribution of this dissertation and is developed to address the research question on how to classify plants at different scales. Results presented in Table 4.2 support this solution and suggest the guided multi-scale data augmentation as an effective way to train CNNs for the fine-grained categorization of plant species when the dataset presents a wide range of plant scale variations in the images. It is a new approach in multi-scale analysis and trains CNN models to become more robust to scale variations. Specific for the plant categorization task, CNNs trained with this novel data augmentation outperform models trained using conventional preprocessing approaches.

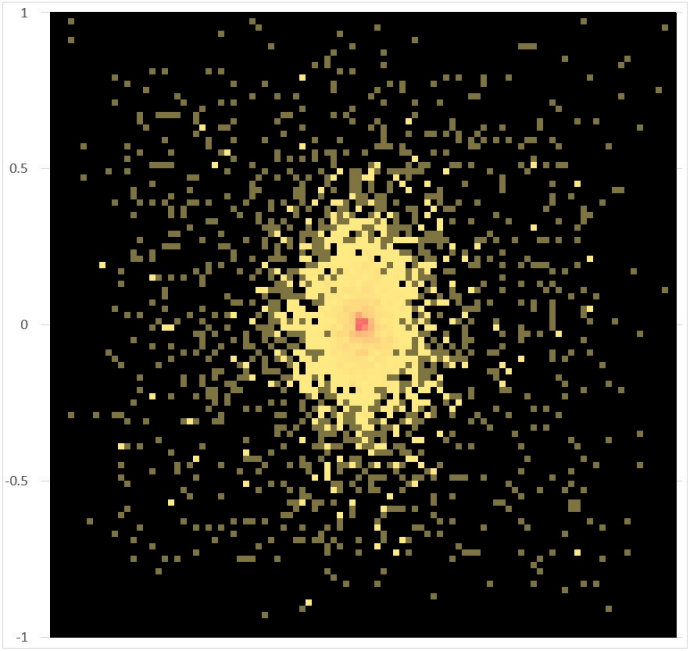


Figure 4.6: Normalized heatmap of 4,778 centroids from UHManoa100 plant image, indicating that plants are more scattered from the center than the BJFU100 dataset images.

4.3 Multi-Scale Classification Process

Previous experiments used one sample from each testing image, extracting only one patch per image for the classification process. These individual testing patches are collected by the same approach used to train the CNNs (Resized, Central Crop, or Bounding Boxes – Figure 3.6). In this Section, I implement the extraction of multi-scale patches not only from the training set but also from images of the testing set. This new classification process combines the analysis of patches from different scales to produce the final species prediction. The following steps summarize this multi-scale classification process:

1. Multi-scale patches are extracted from each input image to perform the classification of the same plant in different scales;

2. The CNN models make predictions for all patches extracted from a single image;

3. The largest average predictive confidence (calculated using the arithmetic mean) indicates the plant species in the image.

This final averaging process helps the models make a more robust prediction when categorizing plants in natural images since the plant is analyzed multiple times over different scales. This new classification process is another contribution of this dissertation and exploits the most representative scales of the image for the plant categorization task.

4.3.1 Pseudocode III

The following pseudocode (Algorithm 4) presents the multi-scale classification process designed to make the *WTPlant* predictions more robust to scale variation of plants. This process outputs the final plant species based on the analysis of multiple patches extracted out of a single input image.

Algorithm 4: MSClassification (I)

Input: Image I
Output: Plant Species S

1 $\mathcal{M}_{Plant}$, $\mathcal{M}_{Flower}$ ← GuidedMSPatchExtraction (I) ;
/* Multi-Scale Classification Process */
2 $Plant_{Prediction}$ ← Average (CNN_{Plant} ($\mathcal{M}_{Plant}$)) ;
3 $Flower_{Prediction}$ ← Average (CNN_{Flower} ($\mathcal{M}_{Flower}$)) ;
/* Combining Plant Organs Predictions */
4 $Final_{Prediction}$ ← ($Plant_{Prediction}$) + ($Flower_{Prediction}$) ;
5 S ← Top-1 Species of $Final_{Prediction}$;
6 **return** S

4.3.2 Experiments (*WTPlant v1.0*)

These experiments use the first version of the *WTPlant* system that implements all the four of the pseudocodes together. The *WTPlant v1.0* trains its CNN models for each target dataset (starting with the BJFU100) and categorizes testing images using the multi-scale classification process.

BJFU100 Dataset

The BJFU100 dataset contains high-resolution images that allow the preprocessing method to collect patches up to 3120x3120 pixels. Using the same CNN models previously trained on the BJFU100 (Table 4.1), I implement the multi-scale classification process using ten patches at different scales per test image. Table 4.3 compares the accuracy results without the multi-scale classification process (Multi-Scale Data Aug.) and implementing it (*WTPlant v1.0*). Presenting a slightly improved performance for this dataset, *WTPlant v1.0* shows that the multi-scale classification process can assist in the categorization of plants even if the dataset has a small scale variation in its images. This new approach utilizes CNNs trained on multi-scale data and also extracts patches at different scales for classification, allowing the *WTPlant* system to perform a more robust analysis of plants in natural images. Therefore, previously trained *ResNets* produced improved accuracy results when analyzing multiple samples of the test images. *ResNet18* remains the most accurate model for this dataset, achieving an accuracy result of 97.80% correct Top-1 classified testing images.

Table 4.3: Results for the BJFU100 dataset with (*WTPlant v1.0*) and without (Multi-Scale Data Aug.) the multi-scale classification process.

CNN Model	Multi-Scale Data Aug.	*WTPlant v1.0*
ResNet18	96.85%	**97.80%**
ResNet34	96.65%	97.58%
ResNet50	91.15%	95.30%

UHManoa100 dataset

As described earlier, experiments using the UHManoa100 dataset have shown improved performance of the *ResNet* models when implementing the new guided multi-scale data augmentation (Table 4.2). This preprocessing method enables the CNNs to learn scale-invariant features, making the classification models more robust to the variation of scale and resolution on this dataset. To further analyze the impact that the multi-scale classification process has during the plant categorization, I purposefully select plant images at various scales for the testing set of this dataset. The difficulties in categorizing these images lead to the creation of the *WTPlant* system and the implementation of the multi-scale classification process specially designed for this type of dataset.

Table 4.4 presents the performance of the plant pipeline of the *WTPlant v1.0* in comparison with previous experiments (Multi-Scale Data Aug.) that analyzes only one patch extracted from testing images. These results show a significant improvement in the models' accuracy when implementing the multi-scale classification process. *ResNet18* still is the most accurate model, improving almost 8% just with the addition of this new process.

In contrast to the results achieved for the BJFU100 dataset (Table 4.3), Table 4.4 shows a significant improvement when the multi-scale classification process is employed. This accuracy boost is mostly because the UHManoa100 dataset presents a wide range of scale variations in its training and testing sets, making the multi-scale classification process necessary for this analysis. Thus, the results presented in this Table show a more significant improvement when compared to BJFU100 experiments.

Table 4.4: Results for the UHManoa100 dataset with (*WTPlant v1.0*) and without (Multi-Scale Data Aug.) the multi-scale classification process.

CNN Model	Multi-Scale Data Aug.	*WTPlant v1.0*
ResNet18	49.28%	**57.19%**
ResNet34	47.84%	56.12%
ResNet50	47.48%	52.16%

Adding the Flower Classification Pipeline

As reported by Wäldchen and Mäder [72], most plant identification approaches rely on shape features to correctly classify leaf images, and methods focusing specifically on flower images generally redirect their analysis from morphological features to textural ones. Implementing different pipelines for separate analysis of plant and flower areas, the *WTPlant* system can train different CNNs to focus on morphological features for the plant classification and textural features for the flower classification. Approaches that rely only on flower images to identify plant species are not very common since not every plant has a flowering stage. This is a strong indicator that a system combining plant and flower analyses may result in a more robust approach. In contrast to existing plant identification systems, *WTPlant* uses a collection of CNNs to classify plants and flowers separately, and then combine their predictions to achieve more accurate results. As shown in Figure 4.1, CNNs output the plant and flower predictions, which are combined to create the final prediction of the plant species (Algorithm 4). This process is called "Prediction Confidence Analysis" and enables the *WTPlant* system to work for both flowering and non-flowering plant species such as ferns, mosses, and liverworts.

For the UHManoa100 dataset, the scene parsing stage of the *WTPlant* system successfully detects the presence of 66 flower species of the 100 analyzed plants. From these flower samples, I select 15 images per species to provide a balanced training set. Consequently, this data augmentation collects 9,900 samples, 10 multi-scale patches from each of the 15 images times 66 species. Most of the 34 plant species whose flowers are not detected do not present a flowering stage during their life cycle. However, some of them do present flowers in their training images, but they are not detected by the scene parsing due to a tiny flower area. In the testing set, only 129 of the 278 test images have flowers detected by the scene parsing approach. For these images, the flower pipeline assists the categorization of the plant species providing additional predictions based on analyzing flowers. Appendix B brings the complete list of analyzed flower species in the UHManoa100 dataset. Even though the flower pipeline does not consider 34 of the 100 plant species, *WTPlant* still categorizes those images using the plant pipeline only.

Table 4.5 presents the performance of each pipeline of the *WTPlant* system (Plant and Flower) as well as the prediction accuracies obtained when these pipelines are combined (*WTPlant v1.0*). In these experiments, *Resnet34* outperforms the other models when analyzing flowers, while *ResNet18* presents the best results for plants. However, when combining the predictions of the *ResNet34* for plants with the *ResNet34* for flowers, the result does not outperform the accuracy of *ResNet18* (for plants only or combined with flowers). Due to the modularity of the *WTPlant* system, different CNN models (in this case, the *ResNet18* for plants and the *ResNet34* for flowers) can work together to predict the final plant species. Consequently, I further improve the *WTPlant v1.0* performance by doing these different classification models work together and achieve the accuracy result of 58.27% when classifying images of the UHManoa100 dataset.

Table 4.5: Individual (Pland and Flower) and combined (*WTPlant v1.0*) accuracy results for the UHManoa100 dataset.

CNN Model	Plant	Flower	***WTPlant v1.0***	***WTPlant v1.0***
ResNet18	**57.19%**	39.50%	57.55%	ResNet18 (plant)+
ResNet34	56.12%	**46.22%**	57.19%	ResNet34 (flower)
ResNet50	52.16%	44.54%	53.24%	**58.27%**

4.4 Incorporating New and Pre-Trained Models

The first version of *WTPlant* utilizes *ResNets* for plant and flower classification. And I use different architectures of this CNN model to evaluate the multi-scale data augmentation and the classification process introduced by this system. The following experiments incorporate classification models implementing inception modules, also called inception models. These modules are an important milestone in the development of state-of-the-art CNNs. Their constant evolution leads to the creation of multiple models, creating different CNN architectures. *WTPlant v2.0* incorporates three of the most popular inception models: *Inception-v3* [66], *Inc-ResNet-v2* [64], and *Xception* [13]. These CNNs implement different inception modules with numerous layers, and, depending on the data, one model may work better than the other. Before the use of these models, most popular CNNs (such as the *ResNets*) seek to pile convolution layers going as deep as possible, hoping to get better performance. On the other hand, inception models use broader architectures to boost performance in terms of both speed and accuracy.

These three CNN models have their first layers bigger than the *ResNet* ones. Because of that, the preprocessing method of the *WTPlant* system extracts larger patches to train these models and extracts ten patches at different scales per image using the same multi-scale data augmentation approach but with close-up patches set to the size of 299x299 pixels. Then I resize all the larger multi-scale patches to this same size and also fit the first layer of the inception models. Similar to *ResNets*, the training process of these new models uses the hyperparameters suggested by their original papers. In this way, *WTPlant v2.0* aggregates the three inception models and trains them to classify plants and flowers using extracted patches.

The search for additional data augmentation approaches and the integration of pre-trained knowledge to assist the training process of these models create new implementation opportunities for the *WTPlant* system. A successful approach to augment images is to use their vertical and horizontal mirrored reflections [55]. As for plants, multi-scaling and vertical mirroring may be the right approaches for data augmentation since it always collects representative plant and flower patches. *WTPlant v2.0* also takes advantage of pre-trained weights to fine-tune its classification models. By starting the training process with previously learned knowledge, inception models incorporated into this version of the system present a considerable boost in accuracy, achieving satisfactory results for the categorization of the UHManoa100 dataset.

Furthermore, I noticed that the most zoomed-in areas (close-up patches) rarely assist the classification process in the plant pipeline. This insight emerged by individually analyzing each scale of patches extracted during the preprocessing method [31]. As an alternative, *WTPlant v2.0* utilizes only the five largest scales and their corresponding mirrored images, balancing the training data for a fair comparison with the previous version of the system. So I use the same number of patches – in this case, ten patches per image – to training the models for this second version of the system, using five multi-scale patches and their mirrored images.

4.4.1 Experiments (*WTPlant v2.0*)

Continuing to focus on the UHManoa100 dataset, I present the *WTPlant v2.0* initial performance in Table 4.6. In these experiments, plant and flower pipelines have six different classification models trained for the analysis of different areas of testing images. *ResNet18* (in the plant pipeline) continues to be the CNN model that better classifies images of the UHManoa100 dataset after 100 epochs, even when compared with inception models. Furthermore, the plant pipeline of this version of the system presents an improved performance when compared with the previous results (Table 4.5). The flower pipeline, however, does not improve its performance and inception models achieved similar results when compared with the previous results. The addition of the flower predictions also helped in most of the cases, outperforming the *ResNet18* plant pipeline to achieve an accuracy of 66.19% when categorizing the UHManoa100 dataset.

Table 4.6: *WTPlant v2.0* accuracy results with CNNs trained for the UHManoa100 dataset.

CNN Model	**Plant**	**Flower**	***WTPlant v2.0***	***WTPlant v2.0***
ResNet18	**65.11%**	42.86%	63.67%	
ResNet34	59.71%	45.38%	60.07%	ResNet18 (plant) +
ResNet50	57.19%	38.66%	56.83%	Inc-ResNet-v2 (flower)
Inception-v3	49.28%	39.50%	50.72%	**66.19%**
Inc-ResNet-v2	62.23%	**46.22%**	64.03%	
Xception	53.24%	38.66%	58.63%	

Not working as expected, inception models did not outperform the *ResNets* in these experiments since overfitting happened during their training processes. In other words, these models learned details of the training images but did not generalize well, negatively impacting the performance of the models when classifying new images. I also implement other data transformation techniques such as rotation, adaptive histogram equalization, and ZCA (Zero-phase Component Analysis) whitening in an attempt to avoid these issues and increase the models' accuracy. However, for the plant categorization problem, none of these preprocessing approaches yield better performance of the inception models.

ImageNet Pre-Trained Weights

Although data augmentation techniques can help to deal with the problem of the limited amount of data to some degree, it may not be enough to provide a good-sized dataset for training deep networks to obtain the best performance. CNN models such as the *Inception-v3* [66], *Inc-ResNet-v2* [64], and the *Xception* [13] have a large number of parameters (up to 54 million), and the lack of training data generally leads to overfitting and poor generalization. Therefore, these CNN models are commonly implemented using pre-trained weights [14]. To fully explore the capability of inception models, I run new experiments using the UHManoa100 dataset to fine-tune these pre-trained models.

The classification models are pre-trained using the ImageNet dataset [58] and fine-tuned for the target dataset using previously learned weights as initial parameter values. In this way, filters learned from datasets such as the ImageNet can be adapted to the classification of plant species. Similar to previous experiments, I perform the fine-tuning of these pre-trained CNNs for 100 epochs. The accuracy results presented in Table 4.7 show that the use of pre-trained models and the fine-tuning process improves the performance more significantly for CNNs with inception modules than the *ResNets*. This phenomenon is natural since the number of parameters in CNNs with inception modules (up to 54 million) is almost twice the amount of parameters of the *ResNets* trained in these experiments (up to 26 million). Therefore the pre-learned weights have more impact on the inception models, making them perform much better on a relatively small dataset such as the UHManoa100.

In particular, the *Inc-ResNet-v2* plant classification model is the most accurate CNN during the experiments. Boosted by the ImageNet pre-trained weights, this model correctly categorizes 89.21% of the testing images. Inception models also present an increase in accuracy for the flower pipeline, and the *Xception* with ImageNet pre-trained weights is the model that outperformed previous results (Tables 4.5 and 4.6). Nevertheless, the merging of both plant and flower pipelines when using the most accurate models does not improve the system's final categorization results. In this case, most of the correctly categorized testing images from the flower pipeline are already correctly classified by the plant pipeline, and the wrong classification of the flower species negatively impacts the final results.

Table 4.7: *WTPlant v2.0* accuracy results with pre-trained CNNs fine-tuned for the UHManoa100.

Pre-Trained CNN	**Plant**	**Flower**	***WTPlant v2.0***	***WTPlant v2.0***
ResNet18	61.51%	48.74%	59.71%	
ResNet34	57.91%	50.42%	57.19%	Inc-ResNet-v2 (plant)+
ResNet50	56.83%	46.22%	55.04%	Xception (flower)
Inception-v3	85.61%	68.91%	84.53%	**87.77%**
Inc-ResNet-v2	**89.21%**	70.59%	87.41%	
Xception	87.05%	**73.11%**	86.69%	

4.5 Graphical User Interface

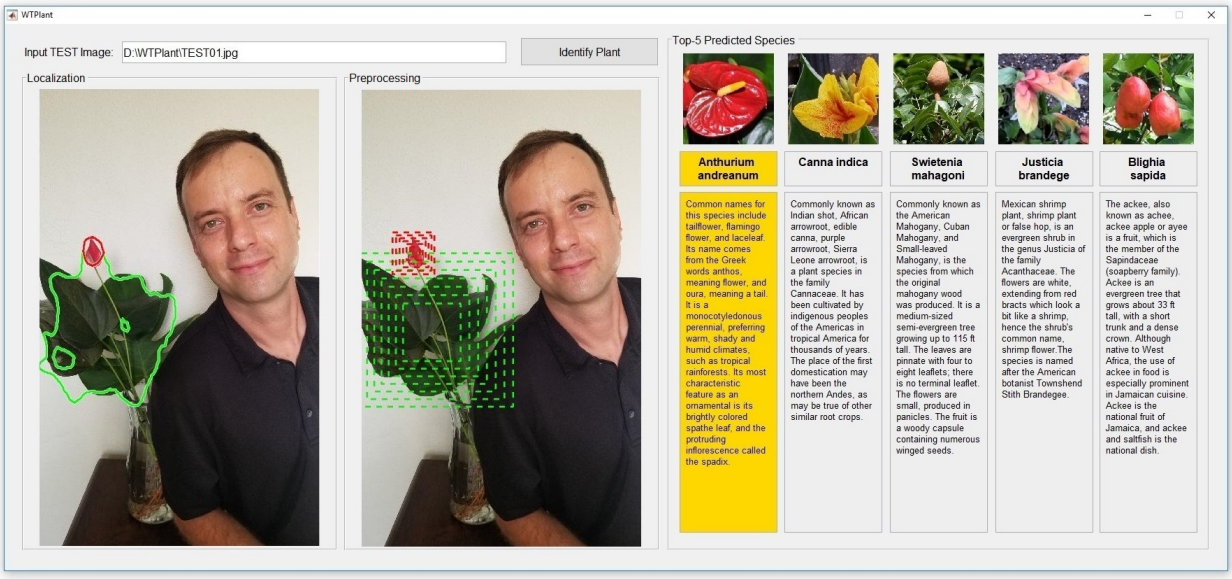
Each stage of the *WTPlant* system (localization, preprocessing, and classification) produces useful information for the plant categorization task. To visualize these stages, I designed a Graphical User Interface (GUI) for the *WTPlant* system. Figure 4.7 presents this GUI, created to demonstrate the *WTPlant* at the International Conference on Multimedia Retrieval (ICMR) [33]. In this GUI, the user inputs the test image and clicks the "Identify Plant" button. *WTPlant* loads the image, and the scene parsing produces plant and flower RoIs. These regions work as a guide to the preprocessing method that extracts multi-scale patches. Plant and flower CNN models classify these patches creating the confidence results for each sample. After the *WTPlant* combines the prediction confidence of each sample for each pipeline, the GUI outputs the Top-5 prediction results with a brief description of the plant taxonomy. I designed this GUI to be a simple and user-friendly interface. However, technical features (such as a variable number of multi-scale patches and their sizes) can be implemented. It is also purely academic, focusing on showing the results of the three main stages of the implemented system. Figure 4.7 (b) shows the result of each one of these stages when the *WTPlant* categorizes a test image. In this case, a plant image with a person partially covering its leaves is categorized. This example simulates what happened during the demonstration at the ICMR, where *WTPlant* successfully categorized live specimens.

4.6 Observations and Discussions

In this chapter, I present a new multi-scale plant categorization system called *WTPlant*. This system exploits the most representative areas of a natural image for the plant categorization task by implementing guided multi-scale approaches and creating a novel data augmentation method. The guided multi-scale data augmentation is a significant contribution of this dissertation, and it trains CNN models to become more robust to scale variations. Experiments detailed in Section 4.2.3 support the hypothesis that this data augmentation method can assist in the training of the classification models. For the plant categorization task, CNNs trained with this novel method outperform models trained using conventional preprocessing approaches.

This chapter also improves the classification process of the implemented system. For that, I present a new multi-scale classification process for the analysis of a test image at various scales, and a pre-selecting process to use the most appropriate multi-scale patches and their mirrored images. *WTPlant v1.0* implements the first new characteristic and improves its classification process. Analyzing the results of BJFU100 dataset experiments (Section 4.3.2), this first version of the system consistently improves in performance when implementing multi-scale approaches. As shown in Table 4.1, over 97% accuracy is achieved when the multi-scale classification process is applied to categorize the BJFU100 images. This result also confirms that I can successfully apply the *WTPlant* system to datasets other than the one created specifically for this research.

(a)



(b)

Figure 4.7: Graphical User Interface (GUI) of the *WTPlant* system. (a) Welcome screen, (b) result screen after plant categorization.

The second version of the *WTPlant* system also applies the multi-scale classification process, but it pre-selects some of the patches. Additionally, the incorporation of inception models and pre-trained weights boosted the accuracy of *WTPlant v2.0* to achieve its best results. Experiments using the UHManoa100 dataset better express the capability of this second version of this system (Section 4.4.1), showing the impact that the improved classification process has on analyzing images at different scales independent of the classification model selected for the task. With the impressive achievement of 89% correctly classified images, the plant pipeline of the *WTPlant v2.0* system best categorize the UHManoa100 dataset when using the *Inc-ResNet-v2* pre-trained classification model. In this case, the addition of the flower pipeline does not improve the overall accuracy. Still, it guarantees the *WTPlant* system's ability to analyze images showing flowers exclusively.

Figure 4.8 presents correctly categorized images illustrating the capability of the *WTPlant* system to handle substantial intra-class scale variation. These plants show a wide range of scale variation, from close-ups of a specific part of the plant to larger scales covering the entire tree. All of them are correctly classified by the *WTPlant v2.0* system when using only the plant pipeline with the *Inc-ResNet-v2* classification model. These images are selected out of the 248 correctly classified ones to represent the improvement on the categorization of plant species when applying the *WTPlant* system. Experiments reported in [31] compare the *WTPlant v2.0* approach with commonly used preprocessing methods (Resizing, Random Crop, and Central Crop), and models trained using these common methods miscategorized plants at different scales. Consequently, images presented in Figure 4.8 show the superiority of the *WTPlant* multi-scale data augmentation and classification process when compared with other approaches.

Figure 4.9 shows examples incorrectly classified by *WTPlant v2.0*. Although these testing images are incorrectly categorized, the final predictions are not far from the correct ones, meaning *WTPlant* lists the right plant species in the Top-3 predictions. It is noticeable that all images in this Figure are close-up pictures with no flowers, showing that the multi-scale approaches may not be effective in categorizing these types of images. Figure 4.10 presents some of the most challenging plant images in the UHManoa100 dataset. These images are incorrectly classified by the best performing CNNs, with the correct plant species not appearing in the Top-5 predictions. The first image is a *Cecropia obtusifolia* with other plant specimens in the background. Because of that, species other than the dominant one also have a strong presence in the extracted patches making it extremely difficult to perform the correct categorization. An image of the *Delonix regia* with other trees mixing up their tops is also one of the most difficult examples in this dataset. Even though its reddish color well distinguishes the target plant, most of it is mixed with the other treetops. Consequently, the CNNs have a hard time distinguishing what the dominant species in the patches present to them is. The third plant is a *Persea americana*, which has its fruits as a dominant characteristic in the image. Currently, *WTPlant* is not prepared for the fine-grained categorization of fruits, which may assist the prediction of plant species for this type of image.

Acacia confusa

Magnolia grandiflora

Tabebuia impetiginosa

Figure 4.8: Examples of correctly categorized plant species showing large scale differences.

Cascabela thevetia

Casuarina equisetifolia

Cupressus sempervirens

Figure 4.9: Examples of incorrectly classified plants but correctly categorized in Top-3 predictions.

Cecropia obtusifolia

Delonix regia

Persea americana

Figure 4.10: Plant images difficult to categorize. Correct species are not in the Top-5 predictions.

CHAPTER 5
EXPANDING THE PLANT CATEGORIZATION SCOPE

The accurate categorization of 100 plant species from multiple datasets (BJFU100 and UH-Manoa100) establishes the *WTPlant* as an effective system for classifying natural images. In this chapter, I discuss the problems associated with expanding the scope of this system and present a solution by modifying the *WTPlant* classification models. The first problem is to define a new target region to list the existing species and collect representative images of each plant species from that environment. By ensuring the correct species annotation, I collect natural images of plants to create a new dataset and define the expanded scope. Considering that this new dataset represents the flora biodiversity of a specific region of the globe, classification models trained over these images compose a plant categorization system with an expanded scope. The second problem is the inclusion of the new species into the *WTPlant* system. A simple solution to this problem would retrain the classification models all over again using the new dataset. Still, previous experiments (Section 4.4.1) show how valuable the pre-trained weights are for the fine-tuning of these models. Therefore, a solution to expand the plant categorization scope of this system must take into account the pre-training process of the classification models. To integrate previously learned knowledge, I implement a modification on the *WTPlant* classification models by replacing the top layers of the CNNs for new extended ones. These new layers accommodate a more significant number of plant species but do not guarantee a high categorization accuracy, creating another implementation problem. In the proposed solution, I also create expert models by training the modified CNNs over domain-specific datasets and use their pre-trained weights to fine-tune the classification models of the system. In this way, knowledge extracted from CNNs pre-trained over plant-related datasets assist in the fine-tuning of the models over the new target dataset. Like this, I expand the plant categorization scope and adapt classification models to inherit powerful discriminative features previously learned during the training over other datasets.

More specifically, this chapter describes approaches taken to deploy the *WTPlant* to a broader environment, expanding its plant categorization scope to 300 species as an example case (UH-Manoa300). Experiments performed using this solution compare the accuracy of the plant pipeline of *WTPlant* system when different datasets (like the iNat682, a plant dataset from *iNaturalist* with 682 species) are used to pre-train its models before the fine-tuning process over the new target dataset. Although it takes much longer to train the CNNs, the resulting models with integrated domain-specific knowledge categorize plants more accurately throughout all experiments. For much larger scopes that encompass more than one environment (e.g., different regions of the globe, continents, or countries), multiple systems can operate in parallel using other guidance methods (such as geolocation of the testing image) to indicate which version to use. In this way, the *WTPlant* can be deployed to larger environments and categorize the entire flora of that specific region.

5.1 Increasing the Number of Plant Species

The first step to increasing the number of plants analyzed by the *WTPlant* system is to gather a representative dataset of the selected species. For this, I used four different approaches to collect natural images of annotated plants:

1. Utilizing the dataset shared by the Botany Department of UHM;
2. Scraping images from the internet (as described in Section 3.3.1);
3. Downloading images from the *iNaturalist*[1] website that are annotated by volunteers[2];
4. Taking high-resolution photographs of live specimens from the UHM campus.

The collected images are organized and reviewed by a UHM botanist specialist[3]. I eliminate images with poor quality and low-resolution (smaller than 400x400 pixels) as well as the incorrectly labeled ones. This process is necessary due to the lack of plant images with annotated species available for the experiments conducted in this study. The combination of multiple sources of natural images and the exhaustive sanity check by a botanist specialist extends the existing dataset of 100 plant species (UHManoa100) to 300 species (UHManoa300).

5.1.1 UHManoa300 Dataset

In an attempt to make the UHManoa300 a more representative dataset, a clean-up process eliminates small, incorrectly annotated images to ensure that only those plants with visible traits are selected. Following this initial process, I organize the UHManoa300 in a collection of 300 plant species with 50 natural images per species, totalizing 15,000 images. With different sizes in height/width varying from 400x400 to 6000x4000 pixels, these images create a diverse dataset representing 300 plant species on many different scales. They also show different plant organs throughout multiple seasons of the year. Consequently, it becomes more difficult to categorize this dataset as the appearance of plants changes considerably across the same species.

Appendix C presents the complete list of plant species selected for the UHManoa300 dataset. Similar to the UHManoa100 (Section 3.3.1), this dataset focuses on common plant species present at the Mānoa Campus of the University of Hawai‘i. Nevertheless, some species are challenging to distinguish from each other, even for the most experienced botanists. As an example, the *Bauhinia spp* listed in this dataset represents a combination of three species: *Bauhinia blakeana*, *Bauhinia purpurea*, *Bauhinia variegata*. For the first two, the only way to tell them apart is that the first is sterile (does not produce pods) and only has purple flowers. At the same time, the second produces

[1] https://www.inaturalist.org – "An online social network of people sharing biodiversity information".
[2] Images classified as "*Research*" on the *iNaturalist* website, usually categorized by botanist specialist volunteers.
[3] https://www.botany.hawaii.edu/faculty/daehler/

seeds (also usually purple flowers but occasionally white or pink flowers). Because of that, the UHManoa300 dataset considers them as one single class called *Bauhinia spp*. Figure 5.1 shows these three species and their similarities. Analogous to *Bauhinia spp*, *Citrus spp* is the union of the species *Citrus reticulata* and *Citrus sinensis* since it is not possible to distinguish them using photos only. UHManoa300 dataset also includes the *Morus spp* pictures as a mixture of *Morus alba* and *Morus rubra*. Another example of merging species is the *Sanchezia speciosa*, which is very difficult to differentiate from *Sanchezia parviflora* (also grown in Hawai'i). Images of these species are generally mixed up and commonly grouped in *spp* classes such as the *Sanchezia spp*.

Figure 5.1: Plant images from the *Bauhinia spp*, unification of three plant species.

The genus *Tabebuia* also has species that are hard to distinguish based on natural images. Therefore, a class called *Tabebuia pink* is created to allocate images from the *Tabebuia heterophylla*, *Tabebuia impetiginosa*, and *Tabebuia rosea* that have similar pink flowers. However, the *Tabebuia* genus also has species such as the *Tabebuia aurea* that present bright yellow flowers and the *Tabebuia berteroi* which has white flowers with wrinkled corolla lobes (both included in the UHManoa300). Other species, such as the *Stigmaphyllon ciliatum* and the *Stigmaphyllon floribundum*, can only be

distinguished by close-up photos of their flowers. Most of the collected images of these two species do not have close-up photos of flowers, so they are not useful for this specific categorization. Hence, UHManoa300 combines these two species in one class called *Stigmaphyllon spp.*

Even though visually indistinguishable species are unified in *spp* classes, the UHManoa300 is still a challenging dataset to classify. Composed of natural images, this dataset is constructed similarly to the UHManoa100, containing plants in different scales, showing multiple organs individually or simultaneously throughout the images, and covering a wide range of image sizes and resolutions. In particular, the *Broussonetia papyrifera* is an interesting species because it has separate male and female plants, and the male flowers look entirely different from the female ones. Species like this create even more challenges for the classification problem forcing the *WTPlant* system to recognize two distinct flowers for one single species. Figure 5.2 presents examples of this species and contrasts the male and female flowers of the plant.

Broussonetia papyrifera (male)

Broussonetia papyrifera (female)

Figure 5.2: Male and female plants of the *Broussonetia papyrifera*.

All images presented in this section belong to the UHManoa300 dataset. They show how complicated categorizing plants in natural images can be, even for the most experienced botanist. To better understand the complexity of the UHManoa300, the species *Acacia koa* is selected to illustrate the diversity of the images in one class. Figure 5.3 shows the central patches of the 50 images of this species selected to be part of the UHManoa300 dataset. Similar to the UHManoa100, different plants may appear in the background or even in front of the dominant plant. And the annotation of the plant in the images indicates the dominant species (*Acacia koa*). Still, the annotated plants cover the most substantial areas of the images. Also shown in this Figure, images in the UHManoa300 dataset contain plants at various scales ranging from leaves and flowers to the entire bush or tree. And original images are in various resolutions, allowing the *WTPlant* system to collect more multi-scale representative training samples than just the ones presented here.

Figure 5.3: All 50 images of the *Acacia koa* in the UHManoa300 dataset.

5.2 Modifying CNN Models to Accommodate Expanded Scope

After constructing a new dataset containing the species in the environment, I adapt the classification models to work with the expanded scope. In this process, top classification layers of the CNNs initially trained to categorize 100 plant species are removed, and the weights of each CNN model are saved without the top layers. For each model, pre-trained weights create a basic knowledge of what the model learned in previous training processes (also called base models). A new and larger classification layer added at the top of a base model creates a new CNN with similar architecture but adapted to work with an extended scope. Thus, knowledge learned from previous experiments can be loaded into the same models but with a larger classification layer at the top. Retraining the modified models over the extracted patches, I fine-tune the CNNs to learn discriminative features between a more significant number of species using the pre-trained weights as a starting point for this process.

The number of new species to be integrated into the scope determine the size of the new classification layer at the top of the model. In this dissertation, I expand the *WTPlant* system categorization from 100 to 300 plant species. In this process, I exclude the two dense layers at the top of the model and replace them with new ones that accommodate the expanded scope. The first one has the same size as the previously excluded layer, but the second one (the last layer) is customized to the exact number of classes in the new dataset (300 plant species). These two top layers work together and are responsible for producing the output predictions of the model. Thereby, modified models have the same architecture as the previously trained ones but are loaded with pre-trained weights and are ready to work with more classes (plant species).

The fine-tuning of modified models over the target dataset updates the parameter values for the entire CNN based on the pre-trained weights. Consequently, well-trained base models lead to a better fine-tuning process of CNNs over the target dataset. Implementing this adaptation over the classification models of the *WTPlant*, I present a solution to expand the scope of this system using multiple pre-trained CNNs. The implemented integration of knowledge from the continuous fine-tuning processes of the CNNs suggests the creation of domain-specific models. These classification models require a higher computational effort to be trained but yield more accurate results.

5.2.1 Integrating Domain-Specific Knowledge in the Plant Pipeline

Due to the lack of training data for most of the fine-grained categorization problems, ImageNet pre-trained weights (Section 4.4.1) are frequently used as initial parameter values during the training process of CNN models. These weights comprise a base model trained over a million images, and this knowledge is useful for most of the visual classification problems. Recently, Cui *et al.* [15], Xiangxi *et al.* [47], and Ngiam *et al.* [48] introduced domain-specific models for fine-tuning their CNNs to different fine-grained categorization problems. Exploring these approaches, I expand the

WTPlant system by adapting its classification CNNs and searching for the best pre-trained weights to fine-tune its models over a new dataset such as the UHManoa300. Experiments using this new target dataset compare the performance of the *WTPlant* system with the knowledge integration of two different plant-related datasets (UHManoa100 and iNat682) using the ImageNet pre-trained weights as a starting point.

ImageNet

Similar to *WTPlant v2.0*, the experiments described in this chapter use the ImageNet [58] pre-trained weights to initialize the training process of the adapted CNN models. These pre-trained weights are directly downloaded and used as base-model for all the following experiments over the UHManoa300 dataset. As suggested by previous experiments on the UHManoa100 dataset (Chapter 4), CNN models implementing inception modules (*Inception-v3* [66], *Inc-ResNet-v2* [64], and *Xception* [13]) take the most advantage of the ImageNet pre-trained weights. These CNN models have millions of parameters and are better fine-tuned over small datasets when pre-trained weights are used as initial parameter values. I collect the next two pre-trained weights (ImageNet+UHManoa100 and ImageNet+iNat682) after the training process of these CNNs using the ImageNet as initial weights. Consequently, the following base models are an integration of previously learned knowledge built over the ImageNet initial weights and what is learned during the fine-tuning process of these models over the UHManoa100 (Section 4.2.3) and the iNat682 datasets.

ImageNet+UHManoa100

The most accurately performing CNNs from previous experiments with 100 plant species (Section 4.4.1) create new base models and the extraction of the ImageNet+UHManoa100 pre-trained weights. As described previously, I remove the top layers of these models and save their weights (parameter values). In this way, knowledge learned during previous experiments can be used during the fine-tuning process of adapted models over the UHManoa300 dataset. It should be noted that the knowledge integration of the ImageNet pre-trained weights fine-tuned over the UHManoa100 dataset creates a domain-specific model that can provide better initial weights for training models on the UHManoa300 dataset.

ImageNet+iNat682

The iNat682 dataset is downloaded from the *iNaturalist* challenge website[4] for the classification of animals and plants. From this dataset, the training process selects only those images from the *Plantae* category, excluding other groups such as animals, insects, fungus, and others. The resulting training set is a highly unbalanced collection of 158,463 images over 682 plant species (and these

[4]https://www.kaggle.com/c/inaturalist-challenge-at-fgvc-2017

images are not part of the UHManoa300). Ranging from 19 to 503, the number of images per class varies according to the endemic nature of each species worldwide. Furthermore, these images vary in resolution, orientation, and focus, making this dataset a very diverse collection of natural images of plants. In an attempt to create a more robust domain-specific CNN model, the experiments using these pre-trained weights aim to integrate the general knowledge from the ImageNet dataset and the plant knowledge from the iNat682 dataset. For this process, the training of the CNN over the iNat682 dataset is initiated with the ImageNet pre-trained weights and performed for 50 epochs. However, no multi-scale patches are extracted from the original images of the iNat682 dataset; only the central crop of each image is used. In such a way, this dataset contributes to the intermediate training process undertaken to create a powerful domain-specific CNN model that learns useful plant-related features for the fine-grained categorization of other plant datasets.

5.2.2 Experiments (*WTPlant v3.0*)

Focusing on the UHManoa300 dataset, the *WTPlant v3.0* expands previous versions of the system to categorize 300 plant species. Due to the clean-up previously performed in this dataset (Section 5.1.1), the preprocessing stage extracts highly representative samples at various scales for plants and flowers. Because of that, training and testing processes on this version of the *WTPlant* system use all the multi-scale extracted patches and their mirrored images. That is, I collect a total of 300,000 (300 plant species $\times$ 50 images per species $\times$ 20 patches per image) patches from the original images. For this balanced dataset, the testing set comprises 10% of the data (five images per species) and the rest is the training set. And I perform the fine-tuning process of the CNNs over the UHManoa300 dataset for 100 epochs. During this process, I divide the training set of extracted patches into training (80%) and validation (20%). It is important to reinforce that, like previous experiments, images selected for the validation set have all their patches for validation only. In this way, I use training and validation patches exclusively in their respective sets for training and validation of the models.

Table 5.1 presents the Top-1 classification accuracies of CNN models pre-trained on different integrated datasets using only the plant pipeline of the *WTPlant* system. In these experiments, the *Xception* outperforms the other models when classifying UHManoa300 images. I further improve its performance by pre-training this model multiple times to integrate domain-specific knowledge. This integration starts with the commonly used ImageNet weights and trains models initially loaded with these parameter values on different plant datasets. It includes pre-trained weights from previous experiments (ImageNet+UHManoa100) and new plant expert models trained with a domain-specific dataset (ImageNet+iNat682). The use of these pre-trained weights allows the *WTPlant v3.0* to improve in accuracy during the categorization of the UHManoa300 plant species. More specifically, the plant pipeline correctly categorized 84% of the testing images when the *Xception* is pre-trained as a plant expert model and used to fine-tune this CNN.

Table 5.1: *WTPlant v3.0* accuracy results with CNNs pre-trained on different dataset for classifying plant species in the UHManoa300 dataset.

CNN model	**ImageNet**	**ImageNet+UHManoa100**	**ImageNet+iNat682**
Inception-v3	75.67%	76.07%	78.80%
Inc-ResNet-v2	76.73%	77.07%	82.33%
Xception	81.20%	81.40%	**84.00%**

As shown in Table 5.1, CNNs fine-tuned for the UHManoa300 achieved more accurate results when pre-trained as domain expert models. Initially, ImageNet pre-trained weights bring a general knowledge with models trained to classify 1,000 common objects. The ImageNet pre-trained models are commonly employed in numerous computer vision problems, but they are limited to the lack of domain-specific knowledge required for fine-grained categorization problems. In the process of creating plant expert models, I use domain-specific datasets to train the CNNs before fine-tuning them over the target dataset (UHManoa300). I collect the ImageNet+UHManoa100 pre-trained weights from CNN models that yielded the best performance in Section 4.4.1. However, CNNs fine-tuned with these pre-trained weights resulted in just slightly more accurate models when compared with no domain-specific knowledge integration (ImageNet). Even though it is a new plant dataset, most UHManoa100 species are present in the UHManoa300 dataset. Hence, knowledge integration is not that evident since the CNN models learned similar discriminating features from UHManoa100 and UHManoa300.

In the process of creating plant expert models, I use a much larger plant dataset to train the CNNs before the fine-tuning process over the target dataset. The training processes over domain-specific datasets help the models to learn more discriminative features and better generalize objects from that domain. Thus, I use natural images of different plant species in the iNat682 dataset to train plant expert CNN models. These models produce domain-specific pre-trained weights that serve as initial parameter values for the fine-tuning process over the UHManoa300. Consequently, the knowledge integration from an extensive dataset such as ImageNet and a domain-specific dataset such as the iNat682 resulted in better pre-trained CNN models for the plant categorization.

5.3 Observations and Discussions

The *WTPlant v3.0* detailed in this chapter is an extended version of the previously presented CNN-based plant categorization system. In this new version, I increase the number of plant species to be categorized from 100 to 300. A botanist helped with the creation and organization of the new dataset (UHManoa300), ensuring a collection of good quality natural images of correctly annotated plants. The adaptation of the classification models to accommodate a larger number of species allows the use of pre-trained weights, improving the models' accuracy. Domain-specific knowledge

from a much larger dataset is integrated into the plant pipeline classification CNNs to create plant expert models. Like this, *WTPlant v3.0* maintains a highly accurate performance similar to previous versions of the system even when expanding its scope. The integration of domain-specific knowledge also helps to avoid the overfitting problem often encountered when training large CNN models over small datasets. Presented in Table 5.1, experimental results using the plant pipeline of the *WTPlant* show the improvement of the system when the CNN models are pre-trained with domain-specific datasets. This accuracy gain is more evident when a large dataset, such as the iNat682, is used to generate the base models. Training over the iNat682 dataset, base models become plant expert CNNs and assist on the fine-tuning of the classification models over the UHManoa300 dataset.

On the other hand, integrate domain-specific knowledge from the iNat682 dataset requires a lot of additional computational effort. Even with the ease of downloading ImageNet pre-trained weights[5], the creation of plant expert models demands the learning of thousands of plant images. For instance, creating plant expert models with the iNat682 dataset required the full training of each CNN over 158,463 images before the final fine-tuning process. With ten multi-scale patches and their mirrored images, the fine-tuning process is also computationally demanding and increases as the system expands. As an example, for the experiments presented in this chapter, I used three GPUs (two *GeForce RTX 2070* and one *GeForce GTX 1080*), and it took almost two months to complete them. Furthermore, I verify the performance of CNN models fine-tuned up to the 50th epoch to predict the gain on performing this process for more than 100 epochs. I observed only a slight average improvement of 0.39% when comparing CNN models fine-tuned for 100 epochs over the other ones. Hence, extending the fine-tuning of the classification models for more than 100 epochs may not be very useful, given the additional computational effort for minimal improvement in accuracy.

Experiments performed in this chapter focus on extending the plant categorization scope from 100 to 300 plant species. State-of-the-art CNN models trained to categorize the UHManoa100 dataset (Chapter 4) achieved highly accurate results, and they serve as a starting point for the experiments with the UHManoa300 dataset. With almost the same number of images per plant species, categorizing 300 plant species can be considered three times more difficult than categorizing 100 species. Fortunately, the adaptation of previously trained CNNs to accommodate an extended classification scope allows the *WTPlant* system to upgrade its models to a new target dataset (UHManoa300). It also creates the opportunity to integrate knowledge from domain-specific datasets, resulting in plant expert models with pre-trained weights (ImageNet+iNat68) that help the fine-tuning process of the classification models. Experimental results presented in Table 5.1 show that CNN models improved their predictions when fine-tuned using domain-specific pre-trained weights. The results also show that the *Xception* [13] is the most effective CNN model for classifying 300 plant

[5]https://keras.io/applications/

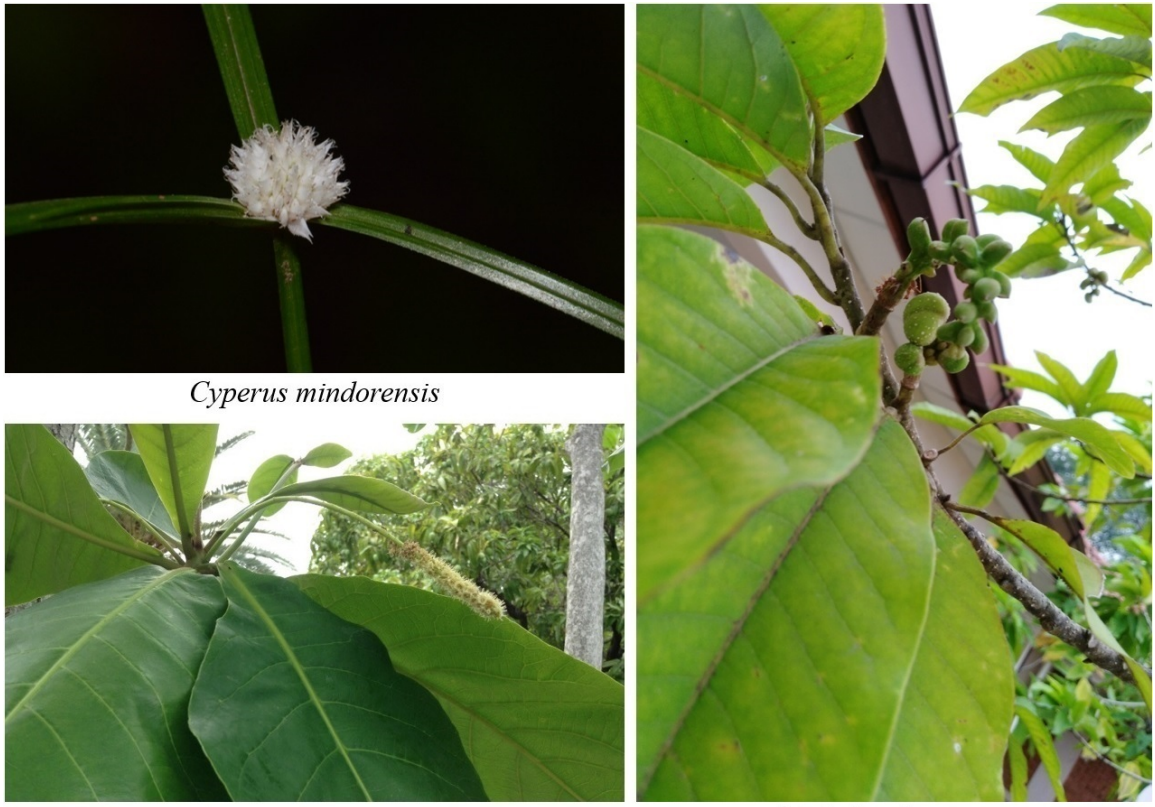
species. Focusing on the predictions of this model, this fine-tuned CNN can correctly categorize an average of 120 testing images more when using the ImageNet+iNat682 pre-trained weights. In particular, ten images are categorized correctly when the *Xception* model uses ImageNet+iNat682 pre-trained weights, and they are not even listed in the Top-5 predictions of models using other pre-trained weights. Figure 5.4 presents some of these images showing possible discriminative features that this CNN (*Xception*) has inherited from its respective plant expert model. Visually reviewing these images, they all resembled the shape of small trees with voluptuous treetops. The iNat682 dataset has multiple annotated images of trees (but it does not include any image from the UHManoa300 testing set) and creates the transferable knowledge of classifying these types of plants. Representative images of entire trees are not common for some species in the UHManoa300 dataset, causing categorization errors. As suggested by experiments performed in this chapter, these errors can be remediated by integrating plant domain-specific knowledge.

Expanding the plant categorization scope brings the challenges of gathering a new target dataset (Section 5.1) and adapt the classification models to work with the new scope (Section 5.2). This chapter addresses both of these challenges, but one problem is noticeable in all versions of the *WTPlant* independently of its scope: The difficulty that this system has on categorizing extreme close-up images. As an example, *WTPlant v2.0* faces problems when categorizing plants in close-up images as the ones presented in Figure 4.9. By considering all the ten multi-scale patches and their mirrored images, this version of the preprocessing method zooms into the plant to extract patches from large scales. And extreme close-up images result in extreme close-up patches that do not cover a representative area of the plant. Figure 5.5 presents some of those incorrectly categorized images showing the same behavior as previous versions of the *WTPlant*. I also noticed that most of the close-up images focus on a specific organ of the plant (fruits or flowers), making it harder for the plant pipeline to collect good representative patches. Therefore, the addition of other pipelines to analyze different organs of the plant may be a suitable alternative to handle those close-up images. Other incorrectly classified images may also take advantage of a separate pipeline for analyzing flowers. As shown in Figure 5.6, some incorrectly categorized plants do not have their plant area visible, while the flowers of that species are evident in the image. In these cases, the *WTPlant v3.0* may successfully categorize the plant species if the flower analysis is incorporated. Therefore, the next chapter focuses on expanding the flower categorization scope and on the improvement when merging plant and flower classification pipelines.

Brugmansia x candida *Catalpa longissima*

Citharexylum spinosum *Erythrina crista-galli*

Figure 5.4: Images correctly categorized by the plant pipeline of the *WTPlant v3.0* system using the *Xception* model with ImageNet+iNat682 pre-trained weights.



Cyperus mindorensis

Terminalia catappa

Michelia champaca

Figure 5.5: Close-up images incorrectly categorized by the plant pipeline of the *WTPlant v3.0*.

Hibiscus rosa-sinensis

Thunbergia grandiflora

Figure 5.6: Flower images incorrectly categorized by the plant pipeline of the *WTPlant v3.0*.

CHAPTER 6
EXPANDING THE FLOWER SCOPE AND MERGING CLASSIFICATION PIPELINES

Flowering plants are the largest and most diverse group in the *Plantae* kingdom, and one of nature's visual delights. Most of the existing flowers are quite colorful and can present themselves in diverse shapes and forms, creating unique traits for the plant species categorization. For this reason, *WTPlant* implements another classification pipeline for the analysis of flowers and combines with the plant pipeline to handle both flowering and non-flowering plants. With the expansion of the plant categorization scope, the number of flower species in the dataset increases accordingly. But not all species listed in the UHManoa300 dataset produce flowers. So for the flower pipeline, I implement the same preprocessing method used for the *WTPlant v3.0* (Chapter 5) to extract multi-scale patches only focusing on the detected flowers. The largest connected flower area segmented by the scene parsing stage (Chapter 3) guides the collection of training and testing patches. In this way, I implement the same preprocessing and fine-tuning methods for both plant (*WTPlant v3.0*) and flower (*WTPlant v3.1*) pipelines for the same target dataset (UHManoa300). However, in this chapter, I consider an unbalanced dataset of flower images to fine-tune the CNNs and verify the efficiency of training with a limited number of patches (less common flower species). And the integration of knowledge from a sizeable domain-specific dataset trains the CNNs to become flower expert classification models. This process creates pre-trained weights to fine-tune the flower classification models of this new pipeline. The presented solution includes a refined analysis of each (plant and flower) prediction confidence to combine the classification scores of multiple expert classification models. This solution also allows the *WTPlant* system to incorporate more classification pipelines and analyze different plant organs such as fruits, barks, seedlings, roots, etc., depending only on the availability of annotated natural images.

6.1 Increasing the Number of Flower Species

Similar to the plant pipeline in Chapter 5, the analysis of flowers starts by preprocessing the training images to extract multi-scale representative patches. Using the UHManoa300 dataset, the preprocessing step of the *WTPlant* extracts ten multi-scale patches from these natural images. However, not all images of this dataset contain flowers. Therefore, the flower pipeline of this system classifies only 246 species (Appendix D) out of the 300 plants. The remaining 54 species do not present flowers in their images or do not have a significant number of flowers detected during the scene parsing stage. As described in Section 3.1.1, the preprocessing stage of the *WTPlant* system may detect many flowers in a single image. Still, it uses the largest area connected to the plant to guide the extraction of multi-scale patches. Images that do not contain any detected flower do not activate the flower pipeline of the system and rely only on the plant classification pipeline.

Unlike earlier versions of the *WTPlant* system, the flower experiments described in this chapter use an unbalanced dataset and try to include all the detected flower species. Nevertheless, I had to leave some species out of the training process. Previous experiments with flowers (Section 4.3.2) considered a threshold of 15 images per flower species to balance the dataset. Although lower thresholds and more flower species have been considered, dealing with an unbalanced dataset is challenging for training a CNN model, especially when there are not enough samples per class to be separated between training and validation. Because of this, the list of analyzed species excludes plants that do not have at least five images with flowers detected. I select this threshold to ensure that at least one image is left for validation (20%) while the others go for the training (80%) for each species.

In summary, the UHManoa300 flower dataset comprises a total of 5,481 images for training (from the 13,500 training images) and 501 for testing (from the 1,500 testing images). Similar to previous experiments, the fine-tuning process of the classification models uses ten multi-scale patches and their mirrored images. In this process, images selected for validation have their patches in the validation set, and the training set is separated using the same procedure. In this way, I ensure that the training, validation, and test patches are mutually exclusive. More specifically, from the total of 109,620 extracted training patches (5,481 images $\times$ 10 multi-scale $\times$ 2 mirrored patches), 88,680 (80.9%) are set as training patches, and 20,940 (19.1%) for validation. As an example, the *Cyperus mindorensis* has only six images with flowers in the training set, resulting in five (>80%) images for training and one (<20%) image for validation. Consequently, the result of this small adjustment reflects on the number of patches selected for training or validation during the fine-tuning process of flower classification models.

6.1.1 Integrating Domain-Specific Knowledge in the Flower Pipeline

After preprocessing the training images and collecting multi-scale patches, the flower pipeline is ready to fine-tune its classification models. As performed for the plant pipeline in *WTPlant v3.0*, flower classification models are pre-trained to integrate knowledge from a much larger dataset. Experimental results in Section 5.2.2 suggest that initial weights pre-trained on a domain-specific dataset help in the creation of plant expert models. Therefore, I employ the same knowledge integration approach for pre-training models over the 2018 FGVCx Flower Classification Challenge[1] dataset. The training process over this dataset creates flower expert models that can be used in the fine-tuning process of the *WTPlant* classification models over a specific flower dataset such as the UHManoa300.

[1] https://www.kaggle.com/c/fgvc2018-flower

ImageNet+FGVC997

Similar to Section 5.2.1, I create flower expert CNN models by fine-tuning the ImageNet pre-trained weights over a large domain-specific dataset. In this case, the dataset used for this process is called FGVC997 and presents 997 different flower species from around the world. This dataset, like the iNat682 (presented in Section 5.2.1), is a highly unbalanced collection of 669,304 natural images of flowers. It ranges from 15 to 3,909 images per species, and varies in size, orientation, and focus, making this dataset a very diverse collection of natural images of flowers. Using the FGVC997 dataset, the training process of flower expert models occurs during 50 epochs, which took weeks to complete it with the available GPUs. As noted earlier (Section 5.3), training CNN models with massive domain-specific datasets is computationally expensive. However, it is an important step to the creation of flower expert models and the integration of domain-specific knowledge. In the end, pre-trained weights from flower expert CNNs work as initial parameter values for the fine-tuning process of the classification models of the system.

6.1.2 Experiments (*WTPlant v3.1*)

Using images from the UHManoa300 dataset, the *WTPlant v3.1* implements the categorization of flowers exclusively. Similar to the plant pipeline in *WTPlant v3.0*, I perform the fine-tuning process of classification models for the flower pipeline during 100 epochs using extracted multi-scale patches. The experiments described below report the categorization results of the flower pipeline only, disregarding images that do not have flowers observed during the scene parsing. With 246 detected species, the flower pipeline has its CNNs adapted to handle this amount of classes using the approach described in Section 5.2. Similar to the plant experiments (*WTPlant v3.0* in Section 5.2.2), I use three pre-trained weights (ImageNet, ImageNet+iNat682, and ImageNet+FGVC997) as initial parameter values for the fine-tuning process over the flower extracted patches.

Table 6.1 presents the accuracy results of CNN models pre-trained on different integrated datasets using only the flower pipeline of the *WTPlant* system. Five hundred one (501) testing images with detected flowers create the testing set used to calculate these accuracies. Similar to the results from plant experiments presented in Table 5.1, CNN models trained to categorize flowers also performed better when integrating domain-specific knowledge. More specifically, the third column of Table 6.1 shows that the ImageNet+iNat682 pre-trained weights for plants (Section 5.2.1) can help (even if slightly) in the fine-tuning of the flower classification models. However, this Table shows that a dataset more closely related to the domain of interest, such as the FGVC997, creates better expert models for the same classification problem. As a result, the *Xception* model that integrates flower-related knowledge (fourth column) outperforms other CNNs with different pre-training strategies, achieving 83.63% accuracy on the classification of the UHManoa300 flowers. On average, the flower classification models improve their performance by 6% when using the ImageNet+FGVC997 pre-trained weights as initial parameter values for the fine-tuning process.

Furthermore, this new version of the system (*WTPlant v3.1*) may assist in the categorization of the plant species (*WTPlant v3.0*) by adding its expert analysis of flowers. The next section describes a way to use both plant and flower extended pipelines and combine their predictions for a more accurate species categorization.

Table 6.1: *WTPlant v3.1* accuracy results for flower images of the UHManoa300 dataset.

CNN model	**ImageNet**	**ImageNet+iNat682**	**ImageNet+FGVC997**
Inception-v3	72.65%	73.85%	80.24%
Inc-ResNet-v2	74.25%	74.85%	81.44%
Xception	78.44%	78.64%	**83.63%**

6.2 Merging Expanded Plant and Flower Pipelines

In contrast to existing plant identification methods, *WTPlant* implements different classification pipelines to categorize plants and flowers, combining their predictions to produce more accurate outputs. As shown in its framework (Figure 4.1), the last step of this system is called "Prediction Confidence Analysis" and is responsible for merging the classification pipelines. It combines the plant and flower analysis by summing the confidence scores of each pipeline (as described in Algorithm 4, in Section 4.3):

$$Final_{Prediction} \leftarrow (Plant_{Prediction}) + (Flower_{Prediction})$$

Experimental results on the UHManoa100 dataset (Tables 4.5 and 4.6) indicate that the combination of the plant and flower predictions may be helpful for the species categorization. However, experiments in Section 4.4.1 combine plant and flower predictions by simple summation. To improve the confidence analysis when merging classification pipelines, I implement a new approach to strengthen the combination of multi-scale plant and flower predictions. In this new approach, I calculate the geometric mean (*Gmean*) to merge the prediction scores of each pipeline ($PlantG_{Prediction}$ and $FlowerG_{Prediction}$). It combines the n prediction values using the product instead of the sum and apply the n-th root instead of the division. The next formula shows how to calculate $PlantG_{Prediction}$ (working similarly for $FlowerG_{Prediction}$):

$$PlantG_{Prediction} \leftarrow \sqrt[n]{(Pred_{patch_1}) \times (Pred_{patch_2}) \times \cdots \times (Pred_{patch_n})}$$

For example, the combination of three multi-scale patches with the predictions of 90%, 20%, and 10% over the wrong species result in 40% using the arithmetic mean and 26.20% using the geometric mean. In this way, the geometric mean consistently handles ratio values and reduces the impact that an incorrect patch prediction has during the categorization process.

Thus, I perform the combination of extracted patches predictions and the merge of both classification pipelines by calculating the geometric mean in this third version of the *WTPlant* system. More specifically, I combine plant and flower predictions using the following new command on line 4 of Algorithm 4:

$$Final_{Prediction} \leftarrow \sqrt{(PlantG_{Prediction}) \times (FlowerG_{Prediction})}$$

For UHManoa300 experiments, geometric mean combines each patch classification score and merge multiple pipelines at the end of the *WTPlant* framework. When merging plant and flower pipelines, the geometric mean combines each array of 246 classes created by the flower pipeline with their respective plant species on the 300 class array. As described in Chapter 3, only flower areas connected to plant (if they exist) activate the flower pipeline. If more than one flower area is detected, only the largest area is further processed. If no plant area is detected, but the flower area is, it activates only the flower pipeline. If no flower and no plant areas are detected, the *WTPlant* system informs that there is "No Plant" in the image. As a result, this multi-pipeline approach enables the categorization of both flowering and non-flowering plant species such as ferns, mosses, and liverworts. And the same approach can be used to expand the categorization to other organs of the plant.

6.2.1 Experiments (*WTPlant v3.2*)

Classification models pre-trained on domain-specific datasets achieved the most accurate results (*WTPlant v3.0* with ImageNet+iNat682 pre-trained weights and *WTPlant v3.1* with ImageNet+ FGVC997 pre-trained weights). Table 6.2 presents the performance of the *WTPlant v3.2* system over the UHManoa300 dataset when plant and flower predictions are combined. A comparison between the previous approach (using *Sum*) and the new combination process (using *Geometric Mean* or *Gmean*) shows the improvement when combining plant and flower predictions using *Gmean* for the categorization of 300 plant species. In both cases (*Sum* and *Gmean*), merging the flower pipeline helped in the categorization of the plants. *Xception* continues to be the most accurate model, correctly classifying 85.53% of testing images (and almost 95% correct with Top-5 predictions).

By comparing the most accurate classification results for plants (Table 5.1), flowers (Table 6.1), and their predictions combined (Table 6.2), I show how the "Prediction Confidence Analysis" step of the *WTPlant* framework improves the system's performance. Although improvement is marginal, experimental results in Table 6.2 show that a categorization method can benefit from analyzing multiple organs of the plant simultaneously. More importantly, the individual analysis of different areas of the image allows the *WTPlant* system to handle diverse natural scenarios showing plants, flowers, or both together. This is often a significant limitation for other plant

categorization approaches that generally focus on a single part of the plant (usually the leaf or the flower). However, it is important to stress that each pipeline of the *WTPlant* framework has to be carefully fine-tuned for its purpose. By integrating domain-specific knowledge from plants and flowers, I train each CNN to become an expert model and use them for the fine-tuning process of the classification models over a target (UHManoa300) dataset. Finally, the plant and flower predictions are combined using the geometric mean to improve the confidence analysis of the system.

Table 6.2: Accuracy results of the *WTPlant v3.2* system combining plant and flower predictions.

CNN model	***WTPlant v3.0*** *Plant Pipeline*	***WTPlant v3.1*** *Flower Pipeline*	***WTPlant v3.2***	
			Using *Sum*	Using *Gmean*
Inception-v3	78.80%	80.24%	78.93%	79.27%
Inc-ResNet-v2	82.33%	81.44%	82.93%	83.33%
Xception	84.00%	83.63%	84.87%	**85.53%**

6.3 Observations and Discussions

After expanding the plant categorization scope in Chapter 5, I face new challenges when adding the flower classification pipeline. As an example, the expanded flower scope creates a highly unbalanced number of patches per species after the preprocessing stage of the *WTPlant* system, that is because not all images from the UHManoa300 dataset contain flowers. Some species do not have flowers at any time of their life cycle, so the flower pipeline automatically excludes them. Also, some species do not have the minimum number of images for the training process. Using this unbalanced dataset of flower images, experiments described in this chapter address the challenge of training and fine-tuning models with limited data per species when expanding the flower categorization scope. Consequently, the *WTPlant* system may help with the analysis of less common species that inevitably present a lack of annotated images.

For experiments with flower images of the UHManoa300 dataset, 37 species (marked in Appendix D) have less than ten flowers detected in their training set. When analyzing these flower species, I notice that most of them have their test images correctly categorized. These results encourage the inclusion of less common plant species, even with a small number of images available. Figure 6.1 presents some of the species that have a small number of training images (less than ten) and have their testing images correctly categorized by the *WTPlant v3.1*. Experimental results with this version of the system support the hypothesis that expert classification models can handle classes with limited training data and learn sufficiently discriminative features.

One particular species (*Calophyllum inophyllum*) represents well the scale variation challenge when expanding the flower categorization scope. The testing images of this species present multiple scales of its flowers during different blooming stages. Figure 6.2 exhibits those that are correctly

Figure 6.1: Less common flower images correctly categorized by the *WTPlant v3.1*.

categorized by the *WTPlant v3.1*. Even though the scale of the flowers differs significantly between the three images, this version of the system can categorize them successfully. With both pipelines functioning (*WTPlant v3.0* and *WTPlant v3.1*), another challenge arises: how to combine plant and flower predictions effectively? Table 6.2 shows that *WTPlant v3.2* can answer this question by properly merging plant and flower pipelines using different functions. The geometric mean (*Gmean*) presents the best results when combining multi-scale predictions of plants and flowers. It also combines each pipeline prediction score to merge the analysis of different organs of the plant, facilitating the inclusion of new pipelines in the future. In this way, this version of the *WTPlant* system can categorize natural images of plants and flowers, combining these and more predictions to perform a robust analysis. Consequently, images where the flower is visible and the plant is

hidden (such as the ones presented in Figure 5.6) are now correctly classified by the system when using its most accurate classification models (*Xception*) for plants and flowers. Figure 6.3 presents other examples of plant images with flowers that are not correctly classified by the plant pipeline but are accurately categorized after the combination with the flower pipeline.

Figure 6.2: Images of the *Calophyllum inophyllum* correctly categorized by the *WTPlant v3.1*.

Experimental results in this Chapter (Table 6.2) and the images in Figure 6.3 show how powerful this method is when expert classification models combine their analysis. Nevertheless, one species in the UHManoa300 dataset did not have any of its test images categorized correctly by the *WTPlant v3.2*. This plant is the *Persea americana* (Avocado Tree) and Figure 6.4 shows the incorrectly categorized images. Although they are misclassified in the Top-1 prediction, *WTPlant* correctly categorize most of them (except the second image in Figure 6.4) in the Top-5, making confusion between this species and the *Artabotrys hexapetalus* plants (Figure 6.5). For this reason, *Persea americana* is considered the most difficult plant species to categorize throughout the experiments of this dissertation. It is noticed that three of the five images shown in Figure 6.4 present its fruits, and the addition of a pipeline for the analysis of this specific plant organ would probably assist in the categorization of this species. Other plant species may also take advantage of a fruit pipeline, enabling the *WTPlant* system to better categorize plants at this stage of their life cycle. Figure 6.6 presents some of these fruitful plants that are incorrectly classified by the most accurate plant and flower models (*Xception*) of the system and would be better categorized if the fruit pipeline is implemented. Despite this, the results presented in Table 6.2 show that even with only two pipelines analyzing two different organs of the plant, the *WTPlant v3.2* system can produce satisfactory results when categorizing an expanded scope of 300 plant species.

Figure 6.3: Images correctly categorized when *WTPlant v3.2* combines plant (*WTPlant v3.0*) and flower (*WTPlant v3.1*) predictions.

Figure 6.4: Images of the *Persea americana* incorrectly categorized by the *WTPlant v3.2*.

Figure 6.5: Images of the *Artabotrys hexapetalus* for comparison with the *Persea americana*.

Citharexylum spinosum

Crescentia cujete

Osteomeles anthyllidifolia

Pithecellobium dulce

Figure 6.6: Images of fruitful species incorrectly categorize by the *WTPlant v3.2*.

CHAPTER 7
WTPLANT MOBILE APPLICATION

The modularity and versatility of the *WTPlant* allow this plant categorization system to work as a classification engine for different applications. As a case study, a partnership with the Harold L. Lyon Arboretum[1] leads to the creation of a new dataset with natural images of local plant species. This new dataset is a collection of 100 plant species living in the 200-acre arboretum located at the upper end of the Mānoa Valley. I use this dataset and previously trained expert CNNs to fine-tune of *WTPlant* classification models for the categorization of the new plant species. To make it available and easy to use for the visitors of the arboretum, I also developed a front-end mobile application that calls the *WTPlant* categorization engine hosted in a remote server to categorize the input image (similar to the Graphical User Interface described in Section 4.5). This mobile version of the *WTPlant* employs an Android Studio template functioning as the front-end for this system. It works by uploading a newly taken or previously saved picture from the visitors' smartphones to a server running the categorization engine fine-tuned explicitly for the plant species present in the arboretum. After the image is categorized, *WTPlant* returns the Top-5 predicted species to the front-end application, showing the probable plant species on the mobile screen. Moreover, different Android Studio templates may work as front-end for any version of the categorization system fine-tuned to other datasets such as the BJFU100, UHManoa100, and UHManoa300. An example of practical use of this system for the Hawaiian community is the beta version of this mobile app tailored to categorize the plant species of a local arboretum and its ecosystem.

7.1 Case Study: Lyon Arboretum App

I officially presented the *WTPlant* system to the Lyon Arboretum board in 2018, and they agreed to participate in this project by providing recently taken photographs of live plant specimens from their ecosystem. These images focus on plant species present in the surroundings of the main trail of the arboretum, aiming to create a plant categorization mobile app that visitors can use while exploring the area. For this beta version of the Lyon Arboretum App, the *WTPlant* system uses only the plant pipeline and extracts fifteen multi-scale patches from the high-resolution images provided. With a powerful categorization engine, the Lyon Arboretum App enables its users to identify, learn, and interact with the flora at both scientific and cultural levels. By increasing the number of plant species in this dataset and expanding the categorization scope, this mobile application will be available for botanists, gardeners, tourists, and the Hawaiian community to enhance their experience when walking through all the trails of the Lyon Arboretum.

[1] https://manoa.hawaii.edu/lyonarboretum/

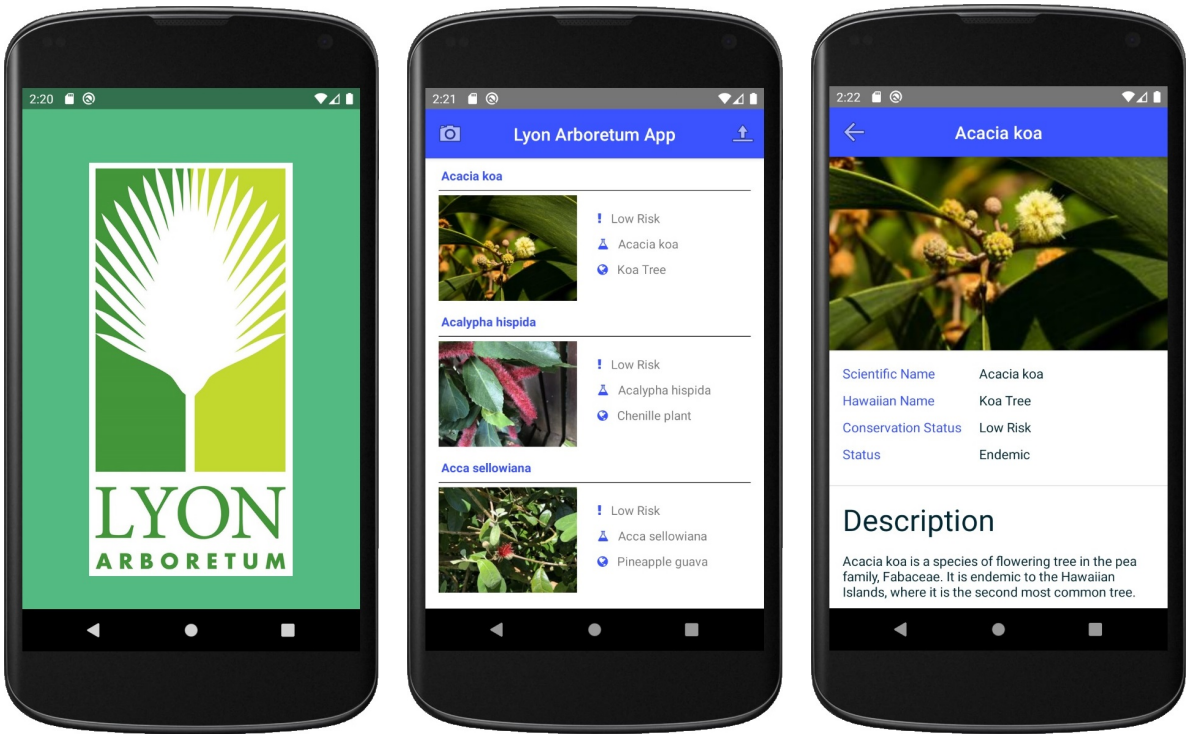
7.1.1 Lyon100 Dataset

The target dataset collected for training the categorization engine of the Lyon Arboretum App is called Lyon100 and contains one hundred species of plants that live in the arboretum. Appendix E presents the complete list of plant species in this dataset. A total of 4,604 images of the same size (3024×4032 pixels) comprise the Lyon100, 4,000 of which are images used for training and the rest for testing. Because the image resolution is high, the *WTPlant* categorization system is set to extract more multi-scale patches totaling 120,000 representative patches for training (100 species × 40 images per species × 15 multi-scale patches × 2 mirrored patches). The increase in the number of multi-scale patches helps avoid overfitting issues and address the lack of training data. However, for the 604 testing images, I use only five patches of the largest scales (as suggested by experiments in Section 4.4 for the plant pipeline) due to the need for a faster categorization process.

7.1.2 Front-End Design

From numerous pre-designed Android Studio templates available online, I select a suitable front-end for the Lyon Arboretum App. This front-end can use numerous templates, and the selected one presents the capability of scroll through the list of plants to facilitate the user's navigation. A detailed description of each species and representative images from the training set of the Lyon100 dataset populate this empty app template. Consequently, users can access all the species information and categorizes plants that they see while walking through the arboretum. They will learn details of each species such as *Scientific Name*, *Hawaiian Name*, *Conservation Status* (Low Risk, Medium Risk, or High Risk), *Status* (Endemic, Indigenous, Introduced, or Invasive), and more. Figure 7.1 presents screenshots from this beta version of the Lyon Arboretum App.

When opening this application on a mobile device, the welcome screen (Figure 7.1(a)) brings the Lyon Arboretum logo for a few seconds until it loads the main screen (Figure 7.1(b)). This second screen brings up all the 100 plant species listed alphabetically, and users can scroll down through the complete list of plant species. Furthermore, users have the option to take new pictures by pressing the top-left camera icon or to load previously taken ones using the top-right upload icon. Either of these two actions uploads a new image to the *WTPlant* system trained to categorize plant species of the Lyon100 dataset. After receiving the Top-5 categorization results back, the front-end lists the predicted plant species presenting them by confidence order (Top-1 to Top-5). At this moment, users have the option of selecting each one of the outputted species and learn more details of the plant, such as its scientific information, history, and common use. The information screen (Figure 7.1(c)) presents these details. As a result, visitors of the Lyon Arboretum can explore the flora of the area and categorize the plant species with self-taken pictures, including selfies with the plants. The deployment of this mobile application may include the geolocation capability, enabling the *WTPlant* system to work as a conservation tool and help with the maintenance of plants.



(a) (b) (c)

Figure 7.1: (a) Welcome screen, (b) Main screen where the user can search through the 100 species, and (c) Information screen providing description, history, and common use of the plant.

7.1.3 Experimental Results

The CNN models of the *WTPlant* achieved satisfactory accuracy results when categorizing plants from the Lyon100 dataset. I calculate these results based on the correct categorization of the 604 testing images. Similar to UHManoa datasets, multiple objects (including other plants and people) are also present in the Lyon100 testing images, making the guidance process of the *WTPlant* system extremely important for this categorization. Using only the plant pipeline, this version of the system integrates previously learned knowledge from plant expert models (Section 5.2.1) for the fine-tuning process over the Lyon100 dataset. Table 7.1 presents the Top-1 and Top-5 accuracy results of these experiments using all the 15 multi-scale testing patches and only the 5 largest scales.

Table 7.1: Accuracy results of the *WTPlant* for the Lyon100 dataset.

CNN model	With 5 largest scales		With 15 multi-scales	
	Top-1	**Top-5**	**Top-1**	**Top-5**
Inception-v3	92.05%	97.35%	91.56%	97.68%
Inc-ResNet-v2	94.70%	98.68%	**94.87%**	98.34%
Xception	93.38%	98.68%	93.71%	98.18%

With almost 95% accuracy, *Inc-ResNet-v2* performs the best for this version of the system, correctly categorizing 573 images from the testing set. The other two CNN models also performed well, achieving similar accuracies. To identify the most challenging images in the Lyon100 dataset, I cross the results from the three CNNs to indicate the images that none of the classification models could identify in their Top-5 predictions. Figure 7.2 presents some of those images.

Alcantarea imperialis *Cordyline fruticosa* *Elaeocarpus angustifolius*

Figure 7.2: Plant images of the Lyon100 dataset that are not correctly classified in Top-5 predictions by all three CNN models used.

First image of Figure 7.2 presents a close-up picture of the *Alcantarea imperialis*. As uncovered in previous experiments, the *WTPlant* system does not perform well when categorizing extreme close-up images. Also, the presence of multiple plants in the background makes this image a hard example to categorize. The second image is a *Cordyline fruticosa* that also has other plants in the background. In this case, the problem is exacerbated due to the open spaces between the plant's leaves, through which other plant species appear. As another extreme example, the third image shows two fruits of the *Elaeocarpus angustifolius* on the ground, incorrectly categorized by the *WTPlant* system. However, a new pipeline for the independent analysis of fruits may help with the categorization of this image.

In this mobile application, the analysis of an image can take more than a minute on the testbed server with one GPU (*GeForce GTX 1080*). To speed up the execution of the Lyon Arboretum App, I extract only five multi-scale patches from the input image and categorize them using the fastest of the three classification models (*Inception-v3*), which reduce the categorization time to about 30 seconds. Currently, I am considering a much faster implementation by implementing the *WTPlant* categorization engine in a cloud computing service such as Amazon Web Services (AWS).

CHAPTER 8
CONCLUSION

In this dissertation, I study the problem of plant species categorization using natural images. Amongst the many challenges of this problem, this dissertation addresses the particular challenges of the plant appearing in different scales by implementing new multi-scale approaches, the analysis of multiple plant organs by using classification pipelines, and the expansion of the plant categorization scope by adapting CNNs and integrating knowledge from domain-specific datasets. Four research questions drive the creation of a solution to this problem. They inquire how to (1) define the most representative areas in the image for the plant categorization task, (2) classify multiple plant organs at different scales, (3) improve the classification process during the categorization of a plant image, and (4) expand the plant categorization scope while maintaining high accuracy.

To answer the question (1), I present a new localization process that identifies the presence of plants and estimates their locations to delimit the most representative areas in the image. This process starts by running a CNN designed to parse scenes from natural environments. It segments the images into multiple regions associated with semantic categories of everyday objects (including plants and flowers). I use the segmented regions to delimit bounding boxes around plants and flowers, defining the most representative areas in the image. This localization process is a necessary step to work with natural images and guide the analysis of plants regardless of their surroundings. In this dissertation, I consider only the largest plant and flower regions to indicate the dominant species in the scene. And the localization process can include the delimitation of smaller plants and flowers also detected during the scene parsing to categorize multiple plant species in the image.

Research question (2) focuses on the classification of multiple plant organs at different scales. I answer this question by extracting multi-scale samples from previously delimited bounding boxes. When used to train the CNNs, I call this process the guided multi-scale data augmentation. And it helps classification models to become more robust to variations on plant size in the image. The multi-scale training process is an innovative data augmentation approach created to improve the learning process of plant and flower images at various scales.

To answer the question (3), I use the same process of extracting multi-scale samples implemented for training in the analysis of a testing image. When used to categorize a new image, I call it the multi-scale classification process. Unlike other categorization methods, this new classification process analyzes multiple areas of the testing images at different scales. For this multi-scale analysis, I combine the predictions of each extracted sample to implement a classification process robust to scale changes. Furthermore, both multi-scale approaches (data augmentation and classification) can contribute to the categorization of different objects that, like plants and flowers, may appear at various scales when captured in nature. Consequently, other categorization problems can take advantage of these novel multi-scale approaches whenever the scale issue is part of the problem.

After conducting numerous experiments over plant datasets with 100 species [31, 32], I answer the question (4) by adapting the classification models to work with an expanded scope of 300 species while using pre-trained weights from expert models to maintain a high categorization accuracy. Initially, a botanist reviews the collected images to correctly annotate this dataset and prepare it for the extraction of plant and flower multi-scale samples. Then I modify the classification models to accommodate a more significant number of plant species and use the extracted samples to fine-tune them. This adaptation allows the expansion to a more significant number of plants until the dataset covers all the species from the target environment. However, additional challenges arose with the expansion of the plant scope, especially to maintain the high categorization accuracy over a more significant number of species. These challenges include the exhaustive collection of plant images correctly annotated, the technical adjustments in the classification models, the creation of domain-specific knowledge, and the proper combination of plant and flower analysis from each classification pipeline. Among them, the creation of expert models by integrating domain-specific knowledge is the most demanding one. I implement this integration process by repeatedly training the classification models over plant-related datasets and extracting their pre-trained weights to integrate knowledge. By dedicating an enormous computational effort to train and fine-tune the modified models, I expand the plant categorization scope to a broader environment while maintaining high accuracy. As a final result, the main objective of this dissertation is achieved, providing an accurate and scalable solution for the problem of plant species categorization using natural images.

Putting together the solutions created to answer the research questions of this dissertation, I present a CNN-base system for the categorization of plant species using natural images called *WTPlant* (What's That Plant?). This system implements a new framework with independent classification pipelines for plants and flowers. On each one of these pipelines, the guidance to extract representative samples comes from different parts of the plant. Therefore, unlike the conventional approach of ensemble models, I implement multiple CNNs working in parallel for more than one classification problem. Both problems (plant and flower classification) count with the multi-scale analysis and combine the prediction of samples collected on each pipeline. In the end, *WTPlant* merges plant and flower pipelines to combine their predictions and output the categorized species.

To evaluate this novel plant categorization system and validate the answers to the research questions, I performed the categorization of multiple plant datasets (UHManoa100, BJFU100, UHManoa300, and Lyon100). As a result, multi-scale methods implemented in the *WTPlant* system improve the accuracy of the classification models substantially when compared with commonly used preprocessing approaches such as resizing and random crop. Additionally, the merging of two independent pipelines enables the *WTPlant* to handle plant images with and without flowers to further improve the plant species categorization. Most of the existing plant categorization applications do not have this capability, being restricted to one or another classification problem at the time.

8.1 Contributions

The main contribution of this research is a novel and expandable CNN-based plant categorization system. By developing the *WTPlant* system, I not only produce a new application to the plant categorization problem but also create valuable scientific knowledge to the analysis of plants in natural images. My contributions include the localization process that performs a scene parsing of the image to identify the most representative areas for the plant categorization task, a new guided multi-scale data augmentation to make the classification models more robust to scale variations, and a new classification process with distinct CNN models for comparative analysis.

Another significant contribution is the comprehensive experimental validation and evaluation with many recently developed CNN architectures performed on the *WTPlant* system over different datasets. These exhaustive experiments indicate a considerable improvement in accuracy when implementing the multi-scale approaches (data augmentation and classification) for the plant categorization task. And the development of this new categorization system brings significant contributions to the fields of Botany. The adaptation and retraining of this system's classification models can redirect its focus to any environment in the world, creating a powerful tool for botanists.

An additional contribution is a process that integrates domain-specific knowledge, which I used to keep the high accuracy of the *WTPlant* system when expanding its scope. This integration process trains the classification models over different plant-related datasets to create expert models. The extraction and publication of pre-trained weights from these plant and flower expert classification models is another contribution of my research. Plant and flower pre-trained weights certainly helped during the fine-tuning process of the *WTPlant* classification models, making the CNNs achieve more accurate results. They are available online[1], and researchers can use them to fine-tune their classification models for new plant datasets.

8.2 Applications

To apply the *WTPlant* multi-scale capability in other systems, I am considering a partnership with existing plant categorization apps. Most of the apps listed in Chapter 2 use CNNs as their classification models and can take advantage of the new multi-scale approaches (data augmentation and classification). Hence, I contacted some of the latest plant categorization apps to offer an association and support to full or partial implementation of the *WTPlant* in their systems. Some of these apps do not publish their results, so I did not list them in the related work. One of these apps is called Seek[2], and developers of this app showed interest in the *WTPlant* system. Designed by the *iNaturalist* team, this mobile app uses images from the iNat682 dataset (and others) to train its classification models. One of the problems with this app is the issue of categorizing plants

[1] https://github.com/jonaskrause/Plant_Flower-Expert_CNN_Models
[2] https://www.inaturalist.org/pages/seek_app

that are far from the observer, and *WTPlant* multi-scale approaches can undoubtedly assist the classification of these plant images. By implementing the guided multi-scale data augmentation and classification processes, I can help solve the scale problem by improving this and most apps that use CNNs as their plant classification models.

With further applications in agriculture, I can apply the *WTPlant* multi-scale approaches together with high-resolution cameras built into field phenotyping systems (mobile platforms for a fast crop phenotyping) to remotely manage each acre of ranches and farms with a close view of the plants. New applications can retrain the *WTPlant* system to work with specific crop plants and use the multi-scale analysis to identify plant abnormalities such as pests, diseases, or invasive species. Consequently, future versions of this system can function as plant growth monitor and identify which crop areas are most fertile for cultivation. Plant counting is also an important task that I can readjust the *WTPlant* system to perform during the crop assessment. These activities are essential to maintain a healthy crop and can lead to a reduction in pesticide use. Thus, an automated multi-scale plant categorization system retrained to work as an agricultural tool may benefit farmers by saving money on crop management and thereby increasing their productivity.

8.3 Future Work

A future work for this research is the categorization of multiple plant species present in the same natural image. I intend to develop this multi-species process not only by considering the application of classification pipelines over small detected areas of plants and flowers but also by creating a multi-location approach to identify clustered plants and flowers. So this future work includes the categorization of entangled plants of different species. During the initial experiments of *WTPlant*, I noticed how difficult it is to segment a plant from a natural background correctly, which led to using the scene parsing only as a guiding process for extracting multi-scale samples. On multiple occasions, the segmentation of the largest plant in the image includes other species in the same delimited area. I can address this problem by implementing a multi-location process to indicate the presence of more than one species of plant in the same area, helping in cases that the segmentation process does not perform a good job. In this new multi-location process, if *WTPlant* detects more than one species by evaluating slightly dislocated close-up samples, classification models can focus exclusively on one part of the identified plant area and output the species present in that specific location. This process will create a map of the image, pointing the location and categorizing all the plant species in the scene. In this way, plant and flower classification models can analyze different details of multiple plants and accurately indicate entangled plant species. However, upgrading the *WTPlant* categorization system to implement this plant mapping feature comes at the high cost of more computational resources and several new research experiments.

APPENDIX A
LIST OF PLANT SPECIES - UHMANOA100 DATASET

Acacia confusa
Acalypha hispida
Alocasia macrorrhiza
Aloe vera
Alpinia purpurata
Anthurium andreanum
Azadirachta indica
Bauhinia variegata
Bixa orellana
Blighia sapida
Bombax glabra
Bougainvillea spp
Brugmansia x candida
Caesalpinia pulcherrima
Calotropis gigantea
Canna indica
Cardamine flexuosa
Cardiospermum grandiflorum
Cascabela thevetia
Cassia bakeriana
Casuarina equisetifolia
Catharanthus roseus
Cattleya spp
Cecropia obtusifolia
Chlorophytum comosum
Clerodendrum quadriloculare
Clitoria ternatea
Cocos nucifera
Codiaeum variegatum
Couroupita guianensis
Crescentia cujete
Crinum asiaticum
Cupressus sempervirens
Cyperus papyrus
Delonix regia
Dendrobium spp
Dichorisandra thyrsiflora
Eichhornia crassipes
Elaeocarpus grandis
Erythrina crista-galli
Eucalyptus deglupta
Eugenia uniflora
Ficus microcarpa
Filicium decipiens
Gardenia brighamii
Gomphrena globosa
Guaiacum officinale
Harpullia pendula
Hedychium coronarium
Hemigraphis alternata
Hibiscus rosa-sinensis
Hippeastrum reticulatum
Impatiens wallerana
Ixora spp
Jasminum sambac
Justicia brandegeana
Kigelia africana
Koelreuteria formosana
Lantana montevidensis
Leea guineensis
Litchi chinensis
Lonicera japonica
Lophostemon confertus
Magnolia grandiflora
Mangifera indica
Metrosideros polymorpha
Musa x paradisiaca
Nandina domestica
Norantea guianensis
Orthosiphon aristatus
Pandanus tectorius
Pentas lanceolata
Persea americana
Petrea volubilis
Phytolacca dioica
Plectranthus scutellarioides
Podranea ricasoliana
Portulacaria afra
Punica granatum
Pyrostegia venusta
Quisqualis indica
Rhaphiolepis umbellata
Solanum seaforthianum
Spathodea campanulata
Stemmadenia littoralis
Strelitzia reginae
Swietenia mahagoni
Symphytum officinale
Tabebuia impetiginosa
Tabernaemontana divaricata
Tamarindus indica
Tectona grandis
Terminalia catappa
Thunbergia battescombei
Tipuana tipu
Tradescantia spathacea
Turnera ulmifolia
Vitex rotundifolia
Waltheria indica
Youngia japonica

APPENDIX B
LIST OF FLOWER SPECIES - UHMANOA100 DATASET

Acacia confusa
Acalypha hispida
Alpinia purpurata
Anthurium andreanum
Bauhinia variegata
Bixa orellana
Blighia sapida
Bougainvillea spp
Caesalpinia pulcherrima
Calotropis gigantea
Canna indica
Cardiospermum grandiflorum
Cassia bakeriana
Catharanthus roseus
Cattleya spp
Clerodendrum quadriloculare
Clitoria ternatea
Codiaeum variegatum
Couroupita guianensis
Delonix regia
Dendrobium spp
Dichorisandra thyrsiflora
Eichhornia crassipes
Erythrina crista-galli
Eugenia uniflora
Gardenia brighamii
Gomphrena globosa
Guaiacum officinale
Harpullia pendula
Hedychium coronarium
Hemigraphis alternata
Hibiscus rosa-sinensis
Hippeastrum reticulatum
Impatiens wallerana
Ixora spp
Jasminum sambac
Justicia brandegeana
Koelreuteria formosana
Lantana montevidensis
Leea guineensis
Litchi chinensis
Magnolia grandiflora
Metrosideros polymorpha
Nandina domestica
Norantea guianensis
Pentas lanceolata
Petrea volubilis
Plectranthus scutellarioides
Podranea ricasoliana
Portulacaria afra
Punica granatum
Pyrostegia venusta
Quisqualis indica
Rhaphiolepis umbellata
Solanum seaforthianum
Spathodea campanulata
Stemmadenia littoralis
Symphytum officinale
Tabebuia impetiginosa
Tabernaemontana divaricata
Thunbergia battescombei
Tipuana tipu
Turnera ulmifolia
Vitex rotundifolia
Waltheria indica
Youngia japonica

APPENDIX C
LIST OF PLANT SPECIES - UHMANOA300 DATASET

Acacia confusa
Acacia koa
Acalypha hispida
Acalypha wilkesiana
Adansonia digitata
Adenanthera pavonina
Agapanthus praecox
Agathis robusta
Ageratum conyzoides
Aleurites moluccana
Allamanda cathartica
Alocasia macrorrhiza
Aloe vera
Alpinia purpurata
Alstonia scholaris
Amaranthus spinosus
Annona muricata
Anthurium andreanum
Araucaria columnaris
Aristolochia littoralis
Artabotrys hexapetalus
Artocarpus altilis
Artocarpus heterophyllus
Asparagus setaceus
Asystasia gangetica
Averrhoa carambola
Azadirachta indica
Azolla filiculoides
Barringtonia asiatica
Bauhinia galpinii
Bauhinia spp
Bidens pilosa
Bixa orellana
Blighia sapida
Bombax ceiba
Bombax glabra
Bougainvillea spp
Brachychiton acerifolium
Breynia distinta
Broussonetia papyrifera
Brugmansia x candida
Brunfelsia latifolia
Caesalpinia ferrea
Caesalpinia pulcherrima
Calliandra calothyrsus
Callistemon citrinus
Callistemon viminalis
Calophyllum inophyllum
Calotropis gigantea
Calyptocarpus vialis
Canna indica
Capsicum frutescens
Cardamine spp
Cardiospermum grandiflorum
Carica papaya
Carissa macrocarpa
Carludovica palmata
Cascabela thevetia
Casimiroa edulis
Cassia bakeriana
Cassia fistula
Cassia x nealiae
Casuarina equisetifolia
Catalpa longissima
Catharanthus roseus
Cecropia obtusifolia
Chloris barbata
Chlorophytum comosum
Citharexylum spinosum
Citrus spp
Clerodendrum quadriloculare
Clitoria ternatea
Clusia rosea
Coccinia grandis
Coccoloba uvifera
Cochlospermum vitifolium
Cocos nucifera
Codiaeum variegatum
Coffea arabica
Colocasia esculenta
Colvillea racemosa
Combretum indicum
Cordia dichotoma
Cordia sebestena
Cordia subcordata
Cordyline fruticosa
Couroupita guianensis
Crescentia cujete
Crinum amabile
Crinum asiaticum
Cupressus sempervirens
Cyperus mindorensis
Cyperus papyrus
Delonix regia
Den Phal Dendrobium
Desmodium spp
Dichorisandra thyrsiflora
Dieffenbachia spp
Dietes bicolor
Dodonaea viscosa
Dracaena marginata
Duranta erecta
Eichhornia crassipes
Elaeocarpus grandis
Elaeodendron orientale

Emilia sonchifolia
Enterolobium cyclocarpum
Epipremnum pinnatum
Erythrina crista-galli
Erythrina sandwicensis
Eucalyptus deglupta
Eugenia uniflora
Euphorbia hirta
Euphorbia milii
Euphorbia pulcherrima
Euphorbia tirucalli
Fagraea berteroana
Ficus carica
Ficus lyrata
Ficus microcarpa
Ficus pseudopalma
Ficus religiosa
Filicium decipiens
Furcraea foetida
Galphimia gracilis
Gardenia brighamii
Gardenia taitensis
Gliricidia sepium
Gomphrena globosa
Gossypium spp(non-native)
Gossypium tomentosum
Graptophyllum pictum
Guaiacum officinale
Harpullia pendula
Hedychium coronarium
Heliconia psittacorum
Hemerocallis lilioasphodelus
Hemigraphis alternata
Hibiscus arnottianus
Hibiscus rosa-sinensis
Hibiscus tiliaceus
Hippeastrum reticulatum
Hiptage benghalensis
Holmskioldia sanguinea
Hura crepitans
Hylocereus undatus
Impatiens wallerana
Ipomoea batatas
Ipomoea horsfalliae
Ixora spp
Jasminum multiflorum
Jasminum sambac
Jatropha integerrima
Justicia betonica
Justicia brandegeana
Kalanchoe pinnata
Kigelia africana
Koelreuteria formosana
Lagerstroemia speciosa
Lantana camara
Lantana montevidensis
Lecythis minor
Leea guineensis
Lemna spp
Leucaena leucocephala
Ligustrum japonicum
Liriope muscari
Litchi chinensis
Lonicera japonica
Lophostemon confertus
Macadamia integrifolia
Macaranga mappa
Macfadyena unguis-cati
Magnolia grandiflora
Malpighia coccigera
Malvastrum coromandelianum
Malvaviscus penduliflorus
Mangifera indica
Manihot esculenta
Manilkara zapota
Melaleuca quinquenervia
Melia azedarach
Merremia tuberosa
Metrosideros polymorpha
Michelia champaca
Mimosa pudica
Monstera deliciosa
Morinda citrifolia
Moringa oleifera
Morus spp
Muehlenbeckia platyclada
Murraya paniculata
Musa x paradisiaca
Mussaenda Queen Sirikit
Myoporum sandwicense
Nandina domestica
Nephrolepis exaltata
Nerium oleander
Norantea guianensis
Nymphaea spp
Ochna thomasiana
Odontonema spp
Olea europaea
Opuntia cochenillifera
Orthosiphon aristatus
Osteomeles anthyllidifolia
Oxalis corniculata
Oxalis debilis
Pandanus tectorius
Pandorea jasminoides
Passiflora edulis
Passiflora foetida
Pentas lanceolata
Pereskia grandifolia
Persea americana
Petrea volubilis
Phymatosorus grossus
Phytolacca dioica
Pilea microphylla
Pimenta dioica
Piper methysticum
Pistia stratiotes
Pithecellobium dulce
Pittosporum tobira
Plantago major

Plectranthus scutellarioides
Plumbago auriculata
Plumbago zeylanica
Plumeria obtusa
Plumeria rubra
Podocarpus macrophyllus
Podranea ricasoliana
Polyscias guilfoylei
Portulaca oleracea
Pouteria sapota
Pritchardia spp
Prosopis pallida
Pseuderanthemum atropurpureum
Pseuderanthemum carruthersii
Pseudobombax ellipticum
Psidium cattleianum
Psidium guajava
Pterocarpus indicus
Pterospermum acerifolium
Punica granatum
Pyrostegia venusta
Ravenala madagascariensis
Rhaphiolepis umbellata
Rhapis excelsa
Ricinus communis
Rivina humilis
Russelia equisetiformis
Saccharum officinarum
Sanchezia spp
Sansevieria trifasciata
Scaevola sericea
Schefflera actinophylla
Sida fallax
Solanum seaforthianum
Sonchus oleraceus
Spathiphyllum x clevelandii
Spathodea campanulata
Spermacoce spp
Stenocarpus sinuatus
Stephanotis floribunda
Sterculia foetida
Stigmaphyllon spp
Strelitzia nicolai
Strelitzia reginae
Swietenia mahagoni
Symphytum officinale
Synedrella nodiflora
Syzygium cumini
Syzygium jambos
Tabebuia aurea
Tabebuia berteroi
Tabebuia pink
Tabernaemontana divaricata
Tabernaemontana litoralis
Tamarindus indica
Tecomanthe dendrophila
Tecomaria capensis
Tectona grandis
Terminalia catappa
Thespesia populnea
Thunbergia battiscombei
Thunbergia erecta
Thunbergia grandiflora
Tipuana tipu
Tradescantia spathacea
Trema orientalis
Trimezia martinicensis
Triplaris surinamensis
Tristellateia australasiae
Turnera ulmifolia
Verbesina encelioides
Vitex rotundifolia
Waltheria indica
Youngia japonica
Zingiber zerumbet

APPENDIX D
LIST OF FLOWER SPECIES - UHMANOA300 DATASET

** Less common flower species in this dataset*

Acacia confusa
Acacia koa
Acalypha hispida
Acalypha wilkesiana
*Adenanthera pavonina**
Agapanthus praecox
Ageratum conyzoides
Aleurites moluccana
Allamanda cathartica
Alpinia purpurata
Anthurium andreanum
Aristolochia littoralis
*Artabotrys hexapetalus**
*Artocarpus heterophyllus**
Asystasia gangetica
Averrhoa carambola
*Azolla filiculoides**
*Barringtonia asiatica**
Bauhinia galpinii
Bauhinia spp
Bidens pilosa
Bixa orellana
Blighia sapida
Bombax ceiba
Bougainvillea spp
Brachychiton acerifolium
Breynia distinta
*Broussonetia papyrifera**
*Brugmansia x candida**
Brunfelsia latifolia
Caesalpinia ferrea
Caesalpinia pulcherrima
Calliandra calothyrsus
Callistemon citrinus
Callistemon viminalis
Calophyllum inophyllum
Calotropis gigantea
Calyptocarpus vialis
Canna indica
Capsicum frutescens
*Cardamine spp**
Cardiospermum grandiflorum
*Carica papaya**
Carissa macrocarpa
*Carludovica palmata**
*Cascabela thevetia**
Cassia bakeriana
Cassia fistula
Cassia x nealiae
*Casuarina equisetifolia**
Catalpa longissima
Catharanthus roseus
*Chloris barbata**
Citharexylum spinosum
Citrus spp
Clerodendrum quadriloculare
Clitoria ternatea
Clusia rosea
Coccinia grandis
Coccoloba uvifera
Cochlospermum vitifolium
Codiaeum variegatum
Coffea arabica
Colvillea racemosa
Combretum indicum
Cordia dichotoma
Cordia sebestena
Cordia subcordata
Cordyline fruticosa
Couroupita guianensis
Crinum amabile
*Cyperus mindorensis**
Delonix regia
Den Phal Dendrobium
Desmodium spp
Dichorisandra thyrsiflora
Dietes bicolor
Dodonaea viscosa
Duranta erecta
Eichhornia crassipes
*Elaeocarpus grandis**
Elaeodendron orientale
Emilia sonchifolia
Erythrina crista-galli
Erythrina sandwicensis
Eugenia uniflora
Euphorbia hirta
Euphorbia milii
Euphorbia pulcherrima
Fagraea berteroana
*Ficus microcarpa**
*Ficus religiosa**
Galphimia gracilis
Gardenia brighamii
Gardenia taitensis
Gliricidia sepium
Gomphrena globosa
Gossypium spp nonnative
Gossypium tomentosum
Graptophyllum pictum
Guaiacum officinale
Harpullia pendula
Hedychium coronarium
Heliconia psittacorum
Hemerocallis lilioasphodelus

Hemigraphis alternata
Hibiscus arnottianus
Hibiscus rosa-sinensis
Hibiscus tiliaceus
Hippeastrum reticulatum
Hiptage benghalensis
Holmskioldia sanguinea
*Hura crepitans**
Hylocereus undatus
Impatiens wallerana
Ipomoea batatas
Ipomoea horsfalliae
Ixora spp
Jasminum multiflorum
Jasminum sambac
Jatropha integerrima
Justicia betonica
Justicia brandegeana
Kalanchoe pinnata
Kigelia africana
Koelreuteria formosana
Lagerstroemia speciosa
Lantana camara
Lantana montevidensis
Lecythis minor
Leea guineensis
*Lemna spp**
Ligustrum japonicum
*Liriope muscari**
Litchi chinensis
Lonicera japonica
*Lophostemon confertus**
*Macadamia integrifolia**
Macfadyena unguis-cati
Magnolia grandiflora
Malpighia coccigera
Malvastrum coromandelianum
Malvaviscus penduliflorus
*Mangifera indica**
*Manilkara zapota**
Melaleuca quinquenervia
Melia azedarach
Merremia tuberosa
Metrosideros polymorpha
Michelia champaca
Mimosa pudica
*Morinda citrifolia**
Moringa oleifera
Morus spp
*Muehlenbeckia platyclada**
Murraya paniculata
*Musa x paradisiaca**
Mussaenda Queen Sirikit
Myoporum sandwicense
Nandina domestica
Nerium oleander
Norantea guianensis
Nymphaea spp
Ochna thomasiana
Odontonema spp
Opuntia cochenillifera
Orthosiphon aristatus
Osteomeles anthyllidifolia
Oxalis corniculata
Oxalis debilis
*Pandanus tectorius**
Pandorea jasminoides
Passiflora edulis
Passiflora foetida
Pentas lanceolata
Pereskia grandifolia
*Persea americana**
Petrea volubilis
*Phymatosorus grossus**
*Phytolacca dioica**
Pimenta dioica
*Pistia stratiotes**
Pithecellobium dulce
Pittosporum tobira
Plectranthus scutellarioides
Plumbago auriculata
Plumbago zeylanica
Plumeria obtusa
Plumeria rubra
Podranea ricasoliana
Portulaca oleracea
*Pouteria sapota**
Pseuderanthemum atropurpureum
Pseuderanthemum carruthersii
Pseudobombax ellipticum
Psidium cattleianum
Psidium guajava
Pterocarpus indicus
Punica granatum
Pyrostegia venusta
Rhaphiolepis umbellata
Ricinus communis
Rivina humilis
Russelia equisetiformis
Sanchezia spp
Scaevola sericea
Sida fallax
Solanum seaforthianum
Sonchus oleraceus
Spathiphyllum x clevelandii
Spathodea campanulata
Spermacoce spp
Stenocarpus sinuatus
Stephanotis floribunda
Sterculia foetida
Stigmaphyllon spp
Strelitzia reginae
Symphytum officinale
Synedrella nodiflora
Syzygium cumini
Syzygium jambos
Tabebuia aurea
Tabebuia berteroi
Tabebuia pink
Tabernaemontana divaricata

Tabernaemontana litoralis
Tecomanthe dendrophila
Tecomaria capensis
*Tectona grandis**
*Terminalia catappa**
Thespesia populnea
Thunbergia battiscombei
Thunbergia erecta
Thunbergia grandiflora
Tipuana tipu
Tradescantia spathacea
*Trema orientalis**
Trimezia martinicensis
Triplaris surinamensis
Tristellateia australasiae
Turnera ulmifolia
Verbesina encelioides
Vitex rotundifolia
Waltheria indica
Youngia japonica
*Zingiber zerumbet**

APPENDIX E
LIST OF PLANT SPECIES - LYON100 DATASET

Acacia koa
Acalypha hispida
Acca sellowiana
Acoelorrhaphe wrightii
Afrocarpus mannii
Ageratum conyzoides
Aglaonema commutatum
Alcantarea imperialis
Alpinia zerumbet
Amherstia nobilis
Aphelandra aurantiaca
Aphelandra sinclairiana
Araucaria columnaris
Averrhoa bilimbi
Bacopa monnieri
Begonia sp
Beilschmiedia anay
Brexia madagascariensis
Brownea coccinea
Brownea hybrida
Brownea macrophylla
Brugmansia x candida
Carex wahuensis
Chamaerops humilis
Chloranthus spicatus
Cibotium glaucum
Clavija nutans
Clerodendrum microstegium
Colocasia esculenta
Cordyline fruticosa
Corypha umbraculifera
Costus dubius
Costus lasius
Couroupita guianensis
Crescentia cujete
Curcuma sp
Cyperus javanicus
Dicorysandra thyrsiflora
Dietes bicolor
Duranta erecta
Elaeocarpus angustifolius
Etlingera coccinea
Etlingera corneri
Heliconia caribaea
Heliconia latispatha
Heliconia magnifica
Heliconia psittacorum
Heliconia rostrata
Heliconia xanthovillosa
Hemerocallis sp
Heterotis rotundifolia
Hibiscus arnottianus
Hippeastrum striatum
Holmskioldia sanguinea
Jatropha multifida
Johannesteijsmannia altifrons
Justicia aurea
Justicia betonica
Liriope muscari
Macaranga tanarius
Magnolia grandiflora
Medinilla magnifica
Metrosideros polymorpha
Metroxylon sp
Microlepia strigosa
Microsorum grossus
Monstera deliciosa
Musa ornata
Myoporum sandwicensis
Myristica fragrans
Osmanthus fragrans
Osmoxylon lineare
Pelagodoxa henryana
Pentagonia macrophylla
Pimenta dioica
Piper magnificum
Pipturus albidus
Plumbago zeylanica
Portlanida grandiflora
Pritchardia martii
Pseudobombax ellipticum
Psydrax odorata
Quesnelia testudo
Renealmia alpinia
Rhapis subtilis
Rhododendron x sp
Sanchezia speciosa
Santalum freycinetianum
Saraca declinata
Spathiphyllum sp
Sphaeropteris cooperi
Sphenomeris chinensis
Strongylodon macrobotrys
Synsepalum dulcificum
Theobroma cacao
Thunbergia mysorensis
Tillandsia cyanea
Warszewiczia coccinea
Widdringtonia schwarzii
Zamia furfuracea

BIBLIOGRAPHY

[1] Antoine Affouard, Hervé Goëau, Pierre Bonnet, Jean-Christophe Lombardo, and Alexis Joly. Pl@ntNet App in the Era of Deep Learning. In *ICLR: International Conference on Learning Representations*, pages 1–6, Toulon, France, 2017.

[2] Bruna Alberton and Leonor Patricia Cerdeira Morellato. Introducing Digital Cameras to Monitor Plant Phenology in the Tropics: Applications for Conservation. *Perspectives in Ecology and Conservation*, 15(2):82–90, 2017.

[3] Md Zahangir Alom, Mahmudul Hasan, Chris Yakopcic, Tarek M. Taha, and Vijayan K. Asari. Improved Inception-Residual Convolutional Neural Network for Object Recognition. *CoRR*, abs/1712.09888, 2017.

[4] Anelia Angelova and Shenghuo Zhu. Efficient Object Detection and Segmentation for Fine-Grained Recognition. In *CVPR*, pages 811–818. IEEE Computer Society, 2013.

[5] Pierre Barre, Ben Stover, Kai Muller, and Volker Steinhage. LeafNet: A Computer Vision System for Automatic Plant Species Identification. *Ecological Informatics*, 40, 2017.

[6] Peter N. Belhumeur, Daozheng Chen, Steven Feiner, David W. Jacobs, W. John Kress, Haibin Ling, Ida Lopez, Ravi Ramamoorthi, Sameer Sheorey, Sean White, and Ling Zhang. Searching the World's Herbaria: A System for Visual Identification of Plant Species. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008*, pages 116–129, Berlin, Heidelberg, 2008. Springer.

[7] Yuri Boykov and Vladimir Kolmogorov. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, 2004.

[8] Steve Branson, Grant Van Horn, Serge J Belongie, and Pietro Perona. Bird Species Categorization Using Pose Normalized Deep Convolutional Nets. *CoRR*, abs/1406.2952, 2014.

[9] Pierre Buyssens, Abderrahim Elmoataz, and Olivier Lézoray. Multiscale Convolutional Neural Networks for Vision-Based Classification of Cells. In *ACCV*, Berlin, Heidelberg, 2012. Springer.

[10] Guillaume Cerutti, Laure Tougne, Julien Mille, Antoine Vacavant, and Didier Coquin. Understanding Leaves in Natural Images - A Model-Based Approach for Tree Species Identification. *Computer Vision and Image Understanding*, 117(10):1482–1501, 2013.

[11] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the Devil in the Details: Delving Deep into Convolutional Nets. *CoRR*, abs/1405.3531, 2014.

[12] Jiahui Chen and Chun Yuan. Convolutional Neural Network Using Multi-scale Information for Stereo Matching Cost Computation. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3424–3428, 2016.

[13] François Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. *CoRR*, abs/1610.02357, 2016.

[14] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge J. Belongie. Large Scale Fine-Grained Categorization and Domain-Specific Transfer Learning. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4109–4118. IEEE Computer Society, 2018.

[15] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge J. Belongie. Large Scale Fine-Grained Categorization and Domain-Specific Transfer Learning. *CoRR*, abs/1806.06193, 2018.

[16] Yin Cui, Feng Zhou, Yuanqing Lin, and Serge J. Belongie. Fine-grained Categorization and Dataset Bootstrapping using Deep Metric Learning with Humans in the Loop. *CoRR*, abs/1512.05227, 2015.

[17] Li Deng and Dong Yu. Deep Learning: Methods and Applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387, 2014.

[18] Hervé Goëau, Pierre Bonnet, Alexis Joly, Vera Bakic, Julien Barbe, Itheri Yahiaoui, Souheil Selmi, Jennifer Carré, Daniel Barthélémy, Nozha Boujemaa, Jean-François Molino, Gregoire Duche, and Aurelien Peronnet. Pl@ntNet Mobile App. pages 423–424, 2013.

[19] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. Multi-scale Orderless Pooling of Deep Convolutional Activation Features. *CoRR*, abs/1403.1840, 2014.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *CoRR*, abs/1512.03385, 2015.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity Mappings in Deep Residual Networks. *CoRR*, abs/1603.05027, 2016.

[22] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A Fast Learning Algorithm for Deep Belief Nets. *Neural Comput.*, 18(7):1527–1554, 2006.

[23] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.

[24] Jing Hu, Zhibo Chen, Meng Yang, Rongguo Zhang, and Yaji Cui. A Multiscale Fusion Convolutional Neural Network for Plant Leaf Recognition. *IEEE Signal Processing Letters*, 25(6), 2018.

[25] Shaoli Huang, Zhe Xu, Dacheng Tao, and Ya Zhang. Part-Stacked CNN for Fine-Grained Visual Categorization. *CoRR*, abs/1512.08086, 2015.

[26] Nursuriati Jamil, Nuril Aslina Che Hussin, Sharifalillah Nordin, and Khalil Awang. Automatic Plant Identification: Is Shape the Key Feature? *IEEE International Symposium on Robotics and Intelligent Sensors (IEEE IRIS2015)*, 76:436–442, 2015.

[27] Alexis Joly, Pierre Bonnet, Antoine Affouard, Jean-Christophe Lombardo, and Hervé Goëau. Pl@ntNet - My Business. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, pages 551–555, New York, NY, USA, 2017. ACM.

[28] Alexis Joly, Pierre Bonnet, Hervé Goëau, Julien Barbe, Souheil Selmi, Julien Champ, Samuel Dufour-Kowalski, Antoine Affouard, Jennifer Carré, Jean-François Molino, Nozha Boujemaa, and Daniel Barthélémy. A Look Inside the Pl@ntNet Experience. *Multimedia Syst.*, 22(6):751–766, 2016.

[29] Alexis Joly, Hervé Goëau, Pierre Bonnet, Vera Bakic, Julien Barbe, Souheil Selmi, Itheri Yahiaoui, Jennifer Carré, Elise Mouysset, Jean-François Molino, Nozha Boujemaa, and Daniel Barthélémy. Interactive Plant Identification Based on Social Image Data. *Ecological Informatics*, 23:22–34, 2014.

[30] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-Scale Video Classification with Convolutional Neural Networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[31] Jonas Krause, Kyungim Baek, and Lipyeow Lim. A Guided Multi-Scale Categorization of Plant Species in Natural Images. In *CVPR Workshop on Computer Vision Problems in Plant Phenotyping (CVPPP 2019)*. IEEE Press, 2019.

[32] Jonas Krause, Gavin Sugita, Kyungim Baek, and Lipyeow Lim. What's That Plant? WTPlant is a Deep Learning System to Identify Plants in Natural Images. In *BMVC Workshop on Computer Vision Problems in Plant Phenotyping (CVPPP 2018)*. BMVA Press, 2018.

[33] Jonas Krause, Gavin Sugita, Kyungim Baek, and Lipyeow Lim. WTPlant (What's That Plant?): A Deep Learning System for Identifying Plants in Natural Images. In *Proceedings of the International Conference on Multimedia Retrieval (ICMR 2018)*. ACM Press, 2018.

[34] Jonathan Krause, Hailin Jin, Jianchao Yang, and Li Fei-Fei. Fine-grained Recognition without Part Annotations. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5546–5555, 2015.

[35] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D Object Representations for Fine-Grained Categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F Pereira, C J C Burges, L Bottou, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[37] 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 W. Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *ECCV (2)*, pages 502–516, 2012.

[38] Mario Lasseck. Image-based Plant Species Identification with Deep Convolutional Neural Networks. In *Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, Dublin, Ireland*, pages 11–14, 2017.

[39] Thi Le, Duc-tuan Tran, and Ngoc-Hai Pham. Kernel Descriptor Based Plant Leaf Identification. In *4th International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–5, 10 2014.

[40] Sue Han Lee, Chee Seng Chan, Simon Mayo, and Paolo Remagnino. How Deep Learning Extracts and Learns Leaf Features for Plant Classification. *Pattern Recognition*, 71:1–13, 2017.

[41] Sue Han Lee, Chee Seng Chan, Paul Wilkin, and Paolo Remagnino. Deep-plant: Plant Identification with Convolutional Neural Networks. *2015 IEEE International Conference on Image Processing (ICIP)*, pages 452–456, 2015.

[42] Fei-Fei Li and Pietro Perona. A Bayesian Hierarchical Model for Learning Natural Scene Categories. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 02*, CVPR '05, pages 524–531, Washington, DC, USA, 2005. IEEE Computer Society.

[43] Liang Lin, Guangrun Wang, Rui Zhang, Ruimao Zhang, Xiaodan Liang, and Wangmeng Zuo. Deep Structured Scene Parsing by Learning with Image Descriptions. *CoRR*, abs/1604.02271, 2016.

[44] Xiao Liu, Tian Xia, Jiang Wang, and Yuanqing Lin. Fully Convolutional Attention Localization Networks: Efficient Attention Localization for Fine-Grained Recognition. *CoRR*, abs/1603.06765, 2016.

[45] Fernand Meyer and Serge Beucher. Morphological Segmentation. *Journal of Visual Communication and Image Representation*, 1(1):21–46, 1990.

[46] Jeff Mo, Eibe Frank, and Varvara Vetrova. Large-scale Automatic Species Identification. In *Proceedings of 30th Australasian Joint Conference on Advances in Artificial Intelligence*, page 301–312. Springer, 2017.

[47] Xiangxi Mo, Ruizhe Cheng, and Tianyi Fang. Pay Attention to Convolution Filters: Towards Fast and Accurate Fine-Grained Transfer Learning. *CoRR*, abs/1906.04950, 2019.

[48] Jiquan Ngiam, Daiyi Peng, Vijay Vasudevan, Simon Kornblith, Quoc V. Le, and Ruoming Pang. Domain Adaptive Transfer Learning with Specialist Models. *CoRR*, abs/1811.07056, 2018.

[49] Maria-Elena Nilsback and Andrew Zisserman. Automated Flower Classification over a Large Number of Classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

[50] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and Dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[51] Florent Perronnin and Christopher R. Dance. Fisher Kernels on Visual Vocabularies for Image Categorization. In *CVPR*. IEEE Computer Society, 2007.

[52] Michael P. Pound, Jonathan A. Atkinson, Darren M. Wells, Tony P. Pridmore, and Andrew P. French. Deep Learning for Multi-task Plant Phenotyping. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2055–2063, 2017.

[53] Michael P. Pound, Aaron S. Jackson, Adrian Bulat, Georgios Tzimiropoulos, Tony P. Pridmore, Andrew P. French, Alexandra J. Townsend, Darren M. Wells, Erik H. Murchie, Jonathan A. Atkinson, Marcus Griffiths, and Michael H. Wilson. Deep Machine Learning Provides State-of-the-art Performance in Image-based Plant Phenotyping. *GigaScience*, 6(10), 2017.

[54] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, pages 413–420, 2018.

[55] Ilya Reshetouski and Ivo Ihrke. *Mirrors in Computer Graphics, Computer Vision, and Time-of-Flight*, volume 8200. 2013.

[56] Jos B. T. M. Roerdink and Arnold Meijster. The Watershed Transform: Definitions, Algorithms and Parallelization Strategies. *Fundam. Inf.*, 41(1,2):187–228, 2000.

[57] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut": Interactive Foreground Extraction Using Iterated Graph Cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[58] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[59] Asma R. Sfar, Nozha Boujemaa, and Donald Geman. Vantage Feature Frames for Fine-Grained Categorization. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 835–842, 2013.

[60] Wen Shi, Fanman Meng, and Qingbo Wu. Segmentation Quality Evaluation Based on Multi-scale Convolutional Neural Networks. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4, 2017.

[61] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556, 2014.

[62] Oskar Söderkvist. Computer Vision Classification of Leaves from Swedish Trees. Number 3132 in LiTH-ISY-Ex, page 74, 2001.

[63] Yu Sun, Yuan Liu, Wang Guan, and Haiyan Zhang. Deep Learning for Plant Identification in Natural Environment. *Computational Intelligence and Neuroscience*, 2017(7361042):6 pages, 2017.

[64] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *CoRR*, abs/1602.07261, 2016.

[65] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. *CoRR*, abs/1409.4842, 2014.

[66] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. *CoRR*, abs/1512.00567, 2015.

[67] Sarah T. Namin, Mohammad Esmaeilzadeh, Mohammad Najafi, Tim B. Brown, and Justin O. Borevitz. Deep Phenotyping: Deep Learning For Temporal Phenotype/Genotype Classification. *bioRxiv*, 2017.

[68] Ahmad P. Tafti, Fereshteh S. Bashiri, Eric LaRose, and Peggy Peissig. Diagnostic Classification of Lung CT Images Using Deep 3D Multi-Scale Convolutional Neural Network. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 412–414, 2018.

[69] Luke Taylor and Geoff Nitschke. Improving Deep Learning using Generic Data Augmentation. *CoRR*, abs/1708.06020, 2017.

[70] Jordan R. Ubbens and Ian Stavness. Deep Plant Phenomics: A Deep Learning Platform for Complex Plant Phenotyping Tasks. *Frontiers in Plant Science*, 8:1190, 2017.

[71] Catherine Wah, Steve Branson, Pietro Perona, and Serge Belongie. Multiclass Recognition and Part Localization with Humans in the Loop. In *2011 International Conference on Computer Vision*, pages 2524–2531, 2011.

[72] 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.

[73] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[74] Jianxiong Xiao, Krista A. Ehinger, James Hays, Antonio Torralba, and Aude Oliva. SUN Database: Exploring a Large Collection of Scene Categories. *Int. J. Comput. Vision*, 119(1):3–22, 2016.

[75] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017.

[76] Songfan Yang and Deva Ramanan. Multi-scale Recognition with DAG-CNNs. *CoRR*, abs/1505.05232, 2015.

[77] Donggeun Yoo, Sunggyun Park, Joon-Young Lee, and In So Kweon. Multi-scale Pyramid Pooling for Deep Convolutional Representation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 71–80, 2015.

[78] Qiangqiang Yuan, Yancong Wei, Xiangchao Meng, Huanfeng Shen, and Liangpei Zhang. A Multi-Scale and Multi-Depth Convolutional Neural Network for Remote Sensing Imagery Pan-Sharpening. *CoRR*, abs/1712.09809, 2017.

[79] Ruimao Zhang, Liang Lin, Guangrun Wang, Meng Wang, and Wangmeng Zuo. Scene Parsing by Weakly Supervised Learning with Image Descriptions. *CoRR*, abs/1709.09490, 2017.

[80] Yu Zhang, Xiu-Shen Wei, Jianxin Wu, Jianfei Cai, Jiangbo Lu, Viet-Anh Nguyen, and Minh N. Do. Weakly Supervised Fine-Grained Categorization With Part-Based Image Representation. *IEEE Transactions on Image Processing*, 25(4):1713–1725, 2016.

[81] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene Parsing through ADE20K Dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[82] Zixia Zhou, Yuanyuan Wang, Jinhua Yu, Yi Guo, Wei Guo, and Yanxing Qi. High Spatial-Temporal Resolution Reconstruction of Plane-Wave Ultrasound Images With a Multichannel Multiscale Convolutional Neural Network. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 65(11):1983–1996, 2018.