

Final Year Project Report

Plant Species' detection in Lough Corrib using Image Processing and Deep Learning

Elizabeth Burke

A thesis submitted in part fulfilment of the degree of
BSc. (Hons.) in Computer Science with Data Science

Supervisor: Dr Eleni Mangina



UCD School of Computer Science
University College Dublin

April 2019

Project Specification:

Ireland has over 12,200 lakes predominantly located along western coastal areas, a number that is approximately twice the European average. The majority of lakes are very small: 66% are < 1ha in area. In contrast, < 2% of Irish lakes (e.g. Lough Corrib and Lough Mask) account for more than 80% of the total lake surface area in the country. Aquatic invasive species have a wide range of impacts on aquatic biodiversity and ecosystem. Invasive plants are non-native species that establish and spread in their new location, generating a negative impact on the local ecosystem and representing one of the leading causes of the extinction of local species. Lough Corrib is a lake in the west of Ireland. It's the largest lake within the Republic of Ireland and the second largest on the island. It covers 176km² mostly in Co. Galway and a small area in Co. Mayo. There are reported to be 365 islands scattered along the length of Lough Corrib. In 1996 Lough Corrib was designated a Special Area of Conservation for 14 different habitats, now listed on the annex I/II of the EU habitats directive. Some of these include petrifying springs, limestone pavement, bog woodland and orchid-rich calcareous grassland.

A large task is being undertaken for the data collection, identification and access of all flora around Lough Corrib. This collection of data will be used to document flora positions and health around the lake, whilst also offering anyone the opportunity to view the flora of the lake in particular media such as websites, books, handouts, ..., etc. Inland Fisheries Ireland wishes to use the data for automated categorisation and identification of the flora via image processing for automated flora identification via ROV and UAV robots. This will allow them to monitor and maintain the local flora, while also providing them with information in a timely fashion of new undocumented species entering the area, such as the *Lagarosiphon major* (Curly leaved waterweed) which requires yearly maintenance to remove and prevent it from spreading further in the lake. The monitoring of invasive species in lakes is challenging, costly, and on the site sometimes dangerous. There are over a hundred species in the area, which can make identification and documentation difficult. The core aim is to create a system whereby flora can be identified by mobile imagery with a high degree of accuracy using deep learning models.

Within the core tasks of this project, the processing will involve selecting and testing a number of deep learning models, training them with the image dataset and lastly documenting the speed and robustness of the flora identification using the selected deep learning models. In regard to the more advanced tasks of this project, the initial focus is on the development of a classification algorithm based on the given image dataset and the calculation of the accuracy (machine learning classifier; optimisation tools for tuning and depiction of results). The subsequent task involves creating an online system to allow for documentation to be uploaded and classified.

A scientifically categorized flora image dataset has been provided by Inland Fisheries Ireland. The dataset has over 112Gb of RGB images. There are 28,000 images across 380 different types of species already categorized into the scientific name of the plant by a botanist in Inland Fisheries Ireland. The data presents flora captured from an average of four different viewpoints, as well as varying backgrounds, proximities and quantities. There are also varying types of genus, which is a specific type of plant within a more general species family.

Acknowledgements

I would like to sincerely thank my supervisor Dr. Eleni Mangina for all her expertise and guidance throughout this process. Her patience and encouragement, combined with constantly challenging me to better myself and aim higher, truly helped make this project a success. I would also like to thank Mr. Evan O’Keeffe for being so generous in sharing his knowledge at my every request. He was an invaluable resource and an incredible mentoring figure, from whom I learned an unfathomable amount.

I would also like to express my sincerest gratitude to Inland Fisheries Ireland, particularly Mr. Ronan Matson. Without Ronan’s enthusiasm, and Inland Fisheries Ireland’s willingness to partner in this project and so generously share their dataset, this project would literally not have been possible. I feel incredibly lucky to have been the first to work with this fascinating data, I understand how rare a dataset like this is and so I’m appreciative, it was truly a privilege.

Finally, I would like to thank my family and friends for all their support, especially my good friends Ms. Bernie, Mr. Bob, Ms. Foxe, Ms. Farrell, Ms. Nic Conmara and of course, Ms. Thay. A project like this is no small task, so the continued understanding, encouragement and belief they offered kept me striving for brilliance in the face of setbacks, and continuously tackling every challenge with unwavering resolve to prove them right, was an invaluable support.

Abstract

Automatic plant species detection is a challenging and fascinating field for both computer scientists and botanists alike. In this research, a deep learning convolutional neural network (CNN) is trained to perform mobile imagery classification on species of flowers found at lake Lough Corrib in Co. Galway. The dataset of flora-classified RGB images, provided by Inland Fisheries Ireland, underwent significant pre-processing, particularly in relation to background removal and data augmentation. Several models of deep learning CNN, with varying amounts of layers and training methods, are evaluated on this dataset. The best model was then embedded in a web application, creating an online system to allow for new plant images to be uploaded and classified.

Acronym List

- IFI : Inland Fisheries Ireland
- DL : Deep Learning
- ML: Machine Learning
- MLA : Machine Learning Algorithm
- RGB : Red, Green, Blue (colour format)
- NDA : Non-disclosure agreement
- NN: Neural Network
- CNN: Convolutional Neural Network
- FC Layer: Fully Connected Layer

Table of Contents

Project Specification:	2
Acknowledgements	3
Abstract	4
Acronym List	4
Table of Contents.....	5
1 Introduction	6
2 Project Background	7
2.1 Automated plant species identification	7
2.2 Deep Learning	9
2.3 Deep Learning for Plant Identification	11
3 Dataset and Data Preparation	17
3.1 Dataset:.....	17
3.2 Data Preparation:.....	18
3.2.1 Background Removal.....	18
3.2.2 Data Augmentation:	21
4 Software Development and Analysis:	23
5 Detailed Design and Software Implementation:	25
5.1 CNN	25
5.2 Online Classification System App	29
6 Testing/Evaluation	31
7 Conclusions and Future Work	34
7.1 Future Work	34
7.2 Conclusion:.....	35
8 References:	36
9 Appendix:	38

1 Introduction

Plant identification is a task that extends far beyond the traditional role of botanists. It's relevant to a much larger portion of society, from livelihoods such as farmers and biologists to nature lovers and eco-tourists alike. However, current methods of plant identification can be both time-consuming for those with rare expert knowledge, and completely inaccessible to those without it. Recent advances in the fields of computer vision and highly accurate pattern recognition algorithms have created a whole new chapter of potential in the field of automated plant species recognition using image processing and deep learning and the application of those technologies. Accurate automatic plant identification is vital for ecological monitoring and as a result for biodiversity conservation.

The original aims for this project were identified through our dialogue with Inland Fisheries Ireland¹ (IFI). The biggest priority use cases they proposed were the following:

1) **Identify:** The ability to automate categorization and identification of these flora species via image processing would be highly valuable. Currently, manual classification by botanists is both time-consuming and a costly task. Due to limited resources, and a shortage of the necessary expertise, plant species identification often takes weeks and is a time-consuming task to visually match each unknown flora species instance in the wild to a documented classification index. The experts have identified the need for an app that can classify these images to a high degree of accuracy, and in real time, making both the work of the botanists more accurate, while making plant classification more accessible to non-botanists and the general public.

2) **Condition:** Documenting the flora health and species quantities around Lough Corrib, would greatly assist Inland Fisheries Ireland in their efforts to identify threats harming the ecosystem around the lake. Flora health is directly linked to the health of the environment around it. Adverse changes to an environment, such as pollution, chemical damage, or drought, would all be reflected in a general deterioration of plant health. By tracking both the quality and quantity of flora around the lake, valuable insight can be harnessed, and actions taken to investigate/treat any issues with the Lough Corrib environment.

3) **Monitoring:** The ability to monitor both the current native species in the Lough Corrib ecosystem, as well as being alerted to new, undocumented species, would be of great use to Inland Fisheries Ireland. The organisation is particularly interested in non-native, or invasive species, that infiltrate an ecosystem and spread, causing damage to the ecosystem balance and jeopardizing the existence of the native species. Ecosystems are sensitive, particularly to foreign flora which they have not previously been exposed to. A specific flora of interest is *Lagarosiphon Major*, an invasive species originating from South Africa that's threatening the brown trout population in Lough Corrib and damaging the fishing tourist industry in the area. The desired ability to monitor these types of invasive species, and alert IFI in a timely manner, would allow them to take measures to contain or eradicate the invading species, thus protecting the natural environment in its original state.

Within the core tasks of this project, to achieve the above goals, the process will involve selecting and testing a number of deep learning models, training them with the image dataset and lastly documenting the speed and robustness of the flora identification using the selected deep learning models. Regarding the more advanced tasks of this project, the initial focus is on the development of a classification algorithm based on the given image dataset and the calculation of the accuracy. The subsequent task involves creating an online system to allow for documentation to be uploaded and classified.

Originally, the aims of this project also included a feature to further enhance our understanding of the Flora composition around Lough Corrib by creating a map of species using GPS metadata. Unfortunately, the GPS

¹ <https://www.fisheriesireland.ie/>

metadata for the flora image dataset was unavailable, therefore eliminating the possibility of that feature at this time. Consequently, the project focused on the classification algorithm for plant recognition.

This report outlines the process undertaken to deliver the aims above. The first chapter provides a summary of the project background, exploring all related and relevant research already undertaken in the plant identification field. Then the focus of the report is transferred to the dataset itself, including the data preparation with specific emphasis on background removal and data augmentation. The report then outlines the software development design and analysis related to delivering the feature scope outlined above. Following on from that, an introduction to machine learning and deep learning is provided. This includes an analysis of several types of algorithm models and a subsequent evaluation of the results. The final element of this report is the conclusions of the findings of this research, and an outline of the potential future work associated with this project.

2 Project Background

When undertaking this project, it was important to do a comprehensive review of all existing relevant studies in this area. A thorough review was completed, spanning across 20 scientific studies published on the topics of plant identification and invasive species detection using image recognition.

Throughout the literature review process as the first phase of this project and from a data scientist point of view, there are two main goals identified:

- Understand the fundamental process of image recognition.
- Acquire the needed knowledge in the area of plant species identification using deep learning.

There have been many studies carried out on the topic, so in order to best advise how to structure the approach for this project, it was necessary to study the existing research and interpret the accuracy and results deduced by those studies. The performance and best practices of this existing research will then dictate the structured approach to my project, and how best to maximise our algorithm's accuracy results for plant identification.

General Trends in automatic plant identification

There are many different approaches to automatic plant identification and due to the different industries involved, a diverse range of strategies have been attempted. However, despite the diverse strategies applied, within the overall picture for the most accurate identification scores, certain trends emerge.

2.1 Automated plant species identification

Within existing research [1] an evaluation of various strategies has taken place for the automated plant species identification, in order to compare their accuracy and quality of their results. A review of the type of machine learning problem which plant identification actually falls into has been identified as a supervised machine learning classification problem, considering it has the key characteristics of discrete, or definitive categories or class labels (i.e. there are clear cut defined species of plants). Figure 2.1 shows the fundamental steps in supervised image recognition models for plant image-based species identification.

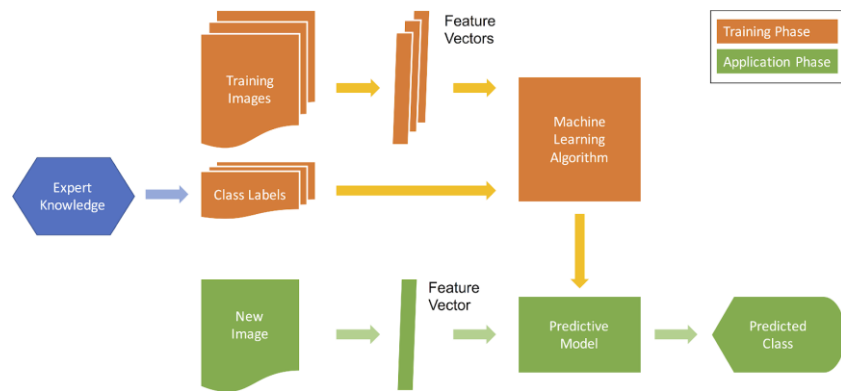


Figure 2.1: Supervised image recognition methodology for plant identification based on images [1]

Following the knowledge acquisition from the experts and the class label identification, the first step in any machine learning algorithm is the training phase. In the case of image identification, it involves training the classifier on both a set of images, as well as their correctly categorized class labels. Once training has been completed, the next step is the application phase, in which we present the trained classifier with new unlabelled images, and the algorithm attempts to correctly assign the image to one of the class labels that we encountered in training.

Another point of interest in the above diagram is ‘Feature Vectors’. In their default state, images are very complex, with millions of pixels and associated colour information. This information is excessively cluttered, and therefore must be reduced by computing the feature vectors. Feature vectors are a filtered down, quantifiable representation of the image that only consists of the most relevant information for solving this particular image identification problem. Research tends to focus on developing feature detection, extraction and developing the methods for automatically computing characteristic feature vectors. Model-free approaches are an alternative to the model-based approaches mentioned above. Their key concept is detecting characteristic interest points and the relevant descriptions using genetic algorithms. Three such examples of these generic algorithms are: scale-invariant feature transform (SIFT), speeded-up robust features (SURF), and histogram of gradients (HOG). These generic algorithms collect visual information in a patch around each interest point as orientation of gradients and have been successfully used for manifold plant classification studies.

Although these feature-based algorithms have been popular in the literature, for the purposes of this project, we are using CNNs as they do not require manually created feature detection and extraction steps. Unlike in the previous models we looked at, both detection and extraction steps are used as an iterative training process, which automatically discovers a statistically suitable image representation (similar to a feature vector) for a given problem. For deep learning models, our core concept is a hierarchical image representation composed of building blocks with increasing complexity per layer. The use of this layered approach has yielded to an unprecedented accuracy of classification results, that had previously been unachievable. Figure 2.2 provides an informative diagram of the existing research [1]. It offers a great visualisation of just how CNNs can classify images of flowers:

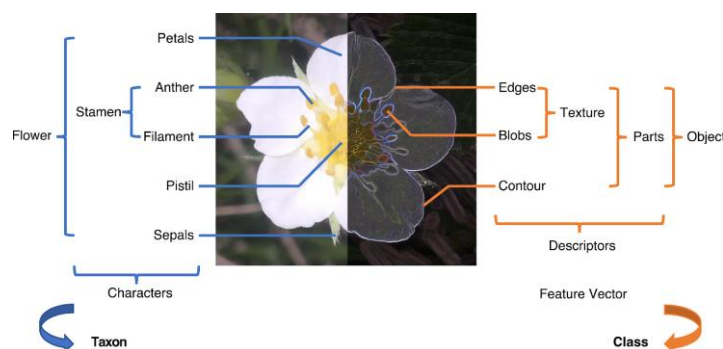


Figure 2.2: Classifying images of flowers with CNN [1]

Image-based taxa identification can be a difficult task for a data scientist without prior botany / geology related knowledge. Below are the main key points from each challenge listed:

1) Large number of taxa to be discriminated: There is a huge number of plant species in the world. Naturally, because of this volume of diversity, the ability to distinguish between a large number of classes/plant species, is subsequently more complicated than distinguishing between just a few. This translates complexity into our classifier creation, as we must provide substantially more training data to achieve accurate results across a large number of species types.

2) Large intraspecific visual variation: Plants that belong to the same species may show significant differences in their morphological characteristics. These differences can vary and fluctuate based on their geographical location and different abiotic factors (e.g. moisture, nutrition, and light condition), their development stage (e.g. differences between a seedling and a fully developed plant), the season (e.g. early flowering stage to a withered flower), and the daytime (e.g. the flower is opening and closing during the day). These changes can affect the entire plant, the leaves, the flowers, etc. There is a lot of contrast across these variables. Figure 2.3 below illustrates the feature variance existing in a single plant, depending on the environmental factors.

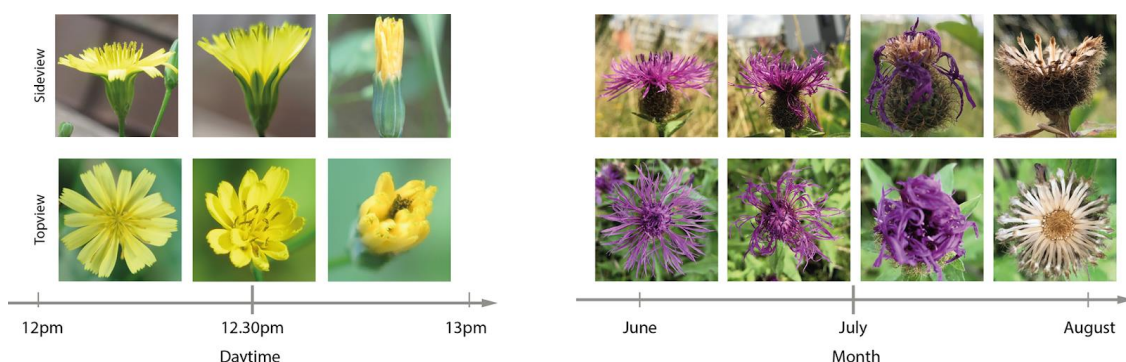


Figure 2.3: Plant feature variance based on environmental factors [1]

3) Small interspecific visual variation: If a species is closely related to each other, they may be extremely similar. This is especially true for the genus, or subclasses within a species family. Even highly experienced botanists find it challenging to identify certain species that can only be distinguished by almost invisible characteristics. We know this is true because, as mentioned when describing the dataset received from IFI earlier in this report, we detailed that there are over 150 classes of species that the botanists could not confidently identify.

4) Rejecting untrained taxa: In addition to being able to match the new unclassified image to the relevant plant species, the classifier must also be able to reject plant images that belong to taxon or species that was not part of the training set. Finding the balance and trade-off point between sensitivity and specificity is a difficult challenge in designing classifiers.

5) Variation induced by the acquisition process: The angle at which you take a picture, as well as additional factors such as the camera quality, focus, brightness, zoom, etc, all impact how the plant species is captured and thus, can potentially distort the appearance of certain features, leading to misclassifications.

2.2 Deep Learning

Deep learning is a specific kind of machine learning. The term machine learning itself refers to the automated detection of meaningful patterns in data [2]. The main principle of machine learning (ML) is the use of learning algorithms. A machine learning algorithm is an algorithm that can learn from data. As outlined in a concise definition of learning by [3], "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

In simpler terms, task T refers to the actual goal we want our algorithm to achieve, so in the context of our project, the task T will be the ability of the algorithm to classify a type of plant species through image recognition into our 31 finite plant species classes. The performance measure P is used to evaluate the ability of the ML algorithm. For classification tasks, the standard metric usually measured is the accuracy of the ML model. Accuracy is referred to as the proportion of the examples for which the model predicts the correct classification output. Evaluation of these performance metrics is completed using a test set of data that has been separated from the data used for the ML algorithm training phase, in order to investigate the performance of the model use-case classifying unseen data. [3]. Experience E, relating to what type of experience/conditions the ML algorithm learns under during the training process, can be categorized as unsupervised or supervised learning. Supervised learning algorithms experience a dataset containing features, but each example is associated with a label. [3] It is called “supervised” because of the presence of the outcome variable to guide the learning process. The dataset provided by Inland Fisheries Ireland for this project has had all data labelled by expert botanists, allowing us to take advantage of supervised learning as our category of experience E for this project. [3].

There are many types of machine learning models that can perform classification tasks. In recent years, deep learning Convolutional Neural Networks (CNNs) have seen a significant breakthrough in computer vision due to the availability of efficient and massively parallel computing on graphics processing units (GPUs) and the availability of large-scale image data necessary for training deep CNNs with millions of parameters [4].

Intro to CNN:

Based on the conclusions from the background research study, CNNs were found to overwhelmingly be the best performing model for plant species identification. Therefore, based on those research findings, the classification model chosen to develop and evaluate for this project is CNNs.

To understand a CNN, it is important to first understand a Neural Network (NN). A Neural Network is a model of self-learning computer program that is created and configured to solve a specific problem. In its most general form, a neural network is a machine that is designed to model the way in which the human brain performs a particular task [5]. NNs consist of a structured network of neurons that are connected, allowing them to communicate with each other to perform certain computations in order to solve a problem.

As mentioned above, given the challenging nature of this project’s plant species image classification task, Convolutional Neural Network (CNN) are the most suitable model to use. CNNs are a specialized kind of neural network for processing data that has a known grid-like topology [3]. For example, in our project the topology data is our plant image data, which can be thought of as a 2-D grid of pixels. The name “convolutional neural network” indicates that the network employs a linear mathematical operation called convolution. Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers [3]. The key characteristic of CNNs is their multi-layer architectures. This involves the inclusion of several convolutional and subsampling layers, potentially including an additional fully connected layer.

The Convolution Layer:

The convolution layer is the main building block of a convolutional neural network, it is comprised of a set of independent convolution neurons. In its most general form, convolution is an operation on two functions of a real-valued argument. This operation is called convolution. The convolution operation is typically denoted with an asterisk [3]:

$$s(t) = (x * w)(t)$$

In convolutional network terminology, the first argument (in this example, the function x) to the convolution is often referred to as the input, and the second (in this example, the function w) as the kernel. The time index is represented by t. The output is sometimes referred to as the feature map.

When convolutions are used on more than one axis at a time, for example in this project, if we use a two-dimensional flower image as our input, we probably also want to use a two-dimensional kernel K :

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(m, n) K(i - m, j - n)$$

Convolutional layers in a CNN model take the input $L \times W \times R$. L represents the length, W represents the width, and R represents the number of channels. The filters in this convolutional layer will be of size $N \times N \times Q$ where N is smaller than the input's dimensions and Q can either be the same value as R or smaller. Each filter is independently convolved with the image, therefore producing a 2-dimensional activation map (also called a feature map). An example is demonstrated below in Figure 2.4.

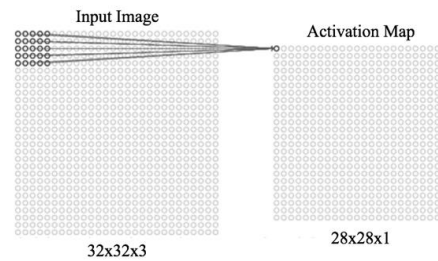


Figure 2.4: Example of Convolution (<https://adeshpande3.github.io/assets/ActivationMap.png>)

The Pooling Layer:

A typical layer of a convolutional network consists of three stages: 1) performing several convolutions in parallel to produce a set of linear activations, 2) running each linear activation through a nonlinear activation function, and 3) using a pooling function to modify the output of the layer. A pooling function replaces the output of the net at a certain location with a summary statistic of the nearby outputs [3]. An example of a popular pooling function is MAX pooling, an operation that reports the maximum output within a rectangular neighbourhood (Figure 2.5).

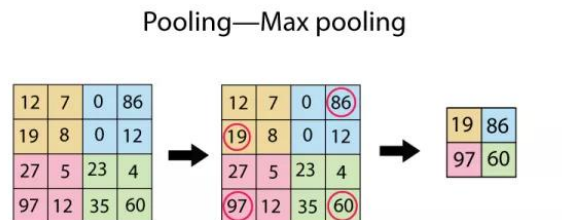


Figure 2.5: Example of MAX pooling (<https://principlesofdeeplearning.com/index.php/2018/08/27/>)

The Fully Connected Layer:

After the feature extraction is performed by our several convolutional layers, we then need to classify the data into various classes. This can be achieved by a fully connected layer. Fully Connected (FC) layers connect every neuron in one layer, to every neuron in another. This process is performed at the end of a model architecture and is usually the last step before the output layer. The FC layer takes an input volume from a convolutional or pooling layer preceding it, and outputs an N dimensional vector where N is the number of classes that the program must choose from. The N number of neurons required for the output layer in this project is 31. Each output neuron corresponding to one of our 31 plant species classes.

2.3 Deep Learning for Plant Identification

As recent computer vision capabilities have progressed, there is a massive research interest in applications of deep learning for plant identification. Current research [6] focuses on the work of plant species' identification in their natural environment using deep learning models. The two key components within this

work are: 1) the type of dataset and 2) the machine learning algorithm. The dataset used is incredibly similar to our own project case study, provided by Inland Fisheries Ireland. This data set, called BJFU100, is collected from natural scenes via images captured on mobile devices. It categorizes over 100 species of ornamental plants in the Beijing Forestry University Campus. While the actual species of Chinese plants classified will vary significantly from our own Irish dataset, there is still a lot to be learned from this research. Particularly, the fact that the data is gathered by mobile device, as that will be our own method of new image collection for testing our own trained classifier model. It is worth noting about this dataset, that the classes in the BJFU dataset need to be all perfectly balanced when designing the classifier. There is exactly 100 images in each category. This removes complications and bias in training the classifiers. Our own raw dataset has some very imbalanced datasets, ranging from 4-30 different pictures depending on the category, which adds to the time consuming task of data preparation of this project, as described in following sections. It will be important that we use data augmentation if necessary, to generate additional training images for categories that may be lacking sufficient quantities of images. It's important to note that this dataset underwent data pre-processing for background removal. Despite the title referencing 'identification in natural environment', the plant images had the backgrounds removed and replaced with just the isolated plant on a plain background.

The model used in this research was the Resnet model as shown in Figure 6. With the network depth increasing in recent advances, traditional methods are not expected to improve the accuracy of classifiers. In fact, adding additional depth can cause problems and a loss of accuracy due to vanishing gradient and degradation. The ResNet residual network introduces skip connections that allow the information (with the input or those learned in earlier layers) to flow more into the deeper layers. Deep residual networks with residual units have shown encouraging accuracy on large scale image recognition tasks.

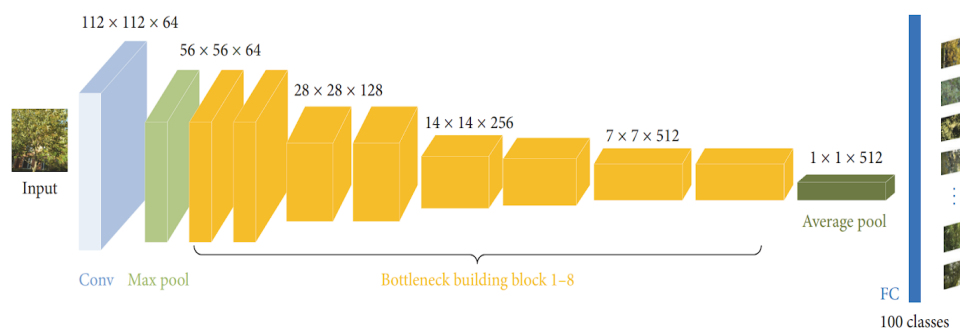


Figure 6: Architecture of Resnet model [6]

1.1.1.1 Summary of background related work

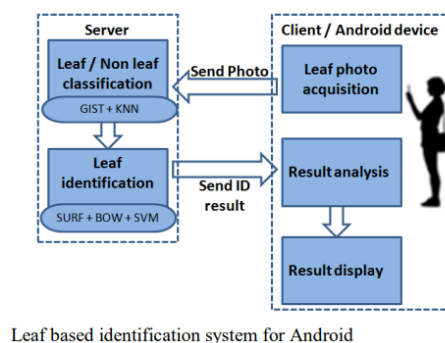
As shown in Table 1, one of the main trends that were found during the research review, was that the vast majority of studies undertaken on plant identification, all focused heavily on leaf-based analysis. In the paper '**LeafNet: A computer vision system for automatic plant species identification**'[7], the study aimed to develop a deep learning system to learn discriminative features from leaf images along with a classifier for species identification of plants. By comparing results with customized systems like LeafSnap, they wanted to show that learning the features by a convolutional neural network (CNN) can provide better feature representation for leaf images compared to hand-crafted features. The outcome of that paper was that when evaluating the recognition accuracies of LeafNet(a CNN-based plant identification system), on the LeafSnap, Flavia and Foliage datasets, the results reveal a better performance of LeafNet compared to hand-crafted customized systems.

Another interesting paper in this domain was the '**Automatic classification of legumes using leaf vein image features**' [8]. There's an interesting re-prioritization of features in this study for segmenting and classifying scanned legume leaves based only on the analysis of their veins is proposed (leaf shape, size, texture, and colour are discarded). This study achieved 87% with the PDA classifier for scanned leaves of soybean, red and white beans. The results are positive, revealing the proposed approach to be an effective and more economical alternative solution which outperforms the manual expert's recognition efforts.

The authors of **'Using Deep Learning for Image-Based Plant Disease Detection'** [9] focus more on the condition of the leaf, rather than the actual leaf itself. This study used a public dataset of 54,306 images of diseased and healthy plant leaves collected under controlled conditions, before training a deep convolutional neural network to identify 14 crop species and 26 diseases (or absence thereof). The trained model achieves an accuracy of 99.35% on a held-out test set, demonstrating the feasibility of this approach. This research will be interesting to try to apply to our project, considering the fact that a public dataset of mobile device images was used here, that is a similar genre of data collection to our own Inland Fisheries Ireland dataset.

Continuing with the research theme of plant disease identification through leaf analysis, an interesting study was done titled **'Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification'** [10]. This paper focuses on disease recognition based on leaf image classification through deep neural networks. The developed research model is capable of recognizing 13 different types of plant diseases out of healthy leaves, with the ability to distinguish plant leaves from their surroundings. Caffe, a deep learning framework, was used to perform the deep CNN training. The experimental results on the developed model achieved precision between 91% and 98%, for separate class tests, on average 96.3%.

One study that adopted a rather unique approach to the leaf identification was titled **'Leaf based plant identification system for Android using SURF features in combination with Bag of Words model and supervised learning'** [11]. This study adopted the interesting strategy of proposing a leaf based plant identification method using SURF features in combination with Bag of Words and supervised learning. This method obtained better results in comparison with other existing methods in the same database. Secondly, this study involved the development of a leaf based plant identification system for Android. The app system of this study is again, particularly relevant to our own project. See below, for a diagram of their app identification system architecture:



Leaf based identification system for Android

Another rather unusual yet interesting approach to identification was **'A Novel Method of Automatic Plant Species Identification Using Sparse Representation of Leaf Tooth Features'** [12]. This research approach was unique as it chose to focus specifically on leaf tooth features. An automatic species identification method using a sparse representation of leaf tooth features was the proposed outcome for this research. Tests on a real-world leaf image dataset show that our proposed method is feasible for species identification.

A paper particularly relevant to the challenges of data pre-processing was **'Data Augmentation for Plant Classification'** [13]. This is important in increasing the number of training images, which in turn improves classifier performances for deep learning. This study evaluates the utility of six individual DA techniques (rotation, blur, contrast, scaling, illumination, and projective transformation) and several combinations of these techniques, resulting in a total of 12 data-augmentation methods. The results show that the CNN methods with particular data augmented datasets yield the highest accuracies, which also surpass previous results on the three datasets. Furthermore, the CNN models trained from scratch profit a lot from data augmentation, whereas the fine-tuned CNN models do not really profit from data augmentation. Finally, we observed that data-augmentation using combinations of rotation and different illuminations or different contrasts helped most for getting high performances with the scratch CNN models. This will be useful in my own project, as we have several unbalanced classes, which we can supplement with augmented data to train a more accurate classifier.

Moving on from leaves, another very popular focus feature for plant identification is perhaps the one most similar to how humans identify plants – the flower. The paper **‘Plant species classification using flower images—A comparative study of local feature representations’** [14] examines methods spanning from detection, extraction, fusion, pooling, to encoding of local features for quantifying shape and colour information of flower images. Findings show large differences among the various studied techniques and that their wisely chosen orchestration allows for high accuracies in species classification. Colour was also found to be an indispensable feature for high classification results, especially while preserving spatial correspondence to grey-level features.

According to the paper **‘Deep machine learning provides state-of-the-art performance in image-based plant phenotyping’** [15], building on artificial neural networks has allowed deep approaches to have many more hidden layers in the network, and hence have greater discriminative and predictive power. This study demonstrates the use of such approaches as part of a plant phenotyping pipeline. It shows the success offered by such techniques when applied to the challenging problem of image-based plant phenotyping and demonstrate state-of-the-art results (>97% accuracy) for root and shoot feature identification and localization. It can fully automate trait identification using deep learning to identify quantitative trait loci in root architecture datasets. This study was particularly interesting as it singles out particular parts of the plant to look for specific features in (i.e. root and shoot), which may be applicable for some of the classes in our Inland Fisheries Ireland dataset, where my initial analysis indicates there are multiple angles of specific plants, often revealing different subsets of plant anatomy/features.

Taking a step back from the more advanced methods of deep learning described above, another somewhat less complex approach is discussed in the paper **‘Plant Classification Using Artificial Neural Networks’** [16]. It involves using a Multi-Layer Perceptron (MLP) artificial neural network trained with a Backpropagation algorithm to perform automatic plant classification. To avoid a data set bias problem, some plant data sets which use different plant features obtained by different feature extraction processes are employed. This study compares the MLP algorithm with several supervised learning methods from plant recognition literature using a statistical hypothesis test of type Friedman/Nemenyi test. The obtained results show the potential of a MLP algorithm to deal with plant classification in an unbiased context. While this particular study focused on leaves, an impressive 97.16% was the best accuracy achieved with the MLP.

For the sake of thoroughness, and in order to broaden the scope of research in the literature review, it was decided to take a step back from plant species classification, and see if there was anything that could be learned from animal species identification. In particular, the focus was on the paper **‘Towards Automatic Wild Animal Monitoring: Identification of Animal Species in Camera-trap Images using Very Deep Convolutional Neural Networks’** [17]. This study used a very deep CNN and reached 88.9% accuracy overall. The main thing learned from this paper was the examples of misclassification, with the most common problem with images being that the illumination was not sufficient to make out key details, or the image was captured too close to the subject, limiting the information available in the picture.

















While the paper **‘Mapping invasive plants using hyperspectral imagery and Breiman Cutler classifications (RandomForest)’** [18], sounds very relevant to my own project, considering the speciality in invasive plants, the study itself was on a very niche type of plant. However, it was interesting how the study builds multiple classification trees by repeatedly taking random subsets of the observational data and using random subsets of the spectral bands to determine each split in the classification trees. The resulting classification trees vote on the correct classification. While not exactly relevant or applicable to my dataset type of RGB images, it was still insightful to learn about the alternative approach taken to invasive species mapping.

One of the most technologically advanced papers researched was **‘UAVs and Machine Learning Revolutionising Invasive Grass and Vegetation Surveys in Remote Arid Lands’** [19]. This paper presents a pipeline process to detect and generate a pixel-wise segmentation of invasive grasses, using buffel grass (*Cenchrus ciliaris*) and spinifex (*Triodia* sp.) as examples. The process integrates unmanned aerial vehicles (UAVs) also commonly known as drones, high-resolution red, green, blue colour model (RGB) cameras, and a data processing approach based on machine learning algorithms. In total, 342,626 samples were extracted

from the obtained data set and labelled into six classes. Segmentation results provided an individual detection rate of 97% for buffel grass and 96% for spinifex, with a global multiclass pixel-wise detection rate of 97%. Obtained results were robust against illumination changes, object rotation, occlusion, background cluttering, and floral density variation.

The final paper of my literature review was '**Visual Classifier for Invasive Plant Species**' [20]. In this project, the aim was to examine effective ways to take on the Kaggle challenge of identifying hydrangea plants in images of forest. The paper evaluates and compares the performance of a VGG-16 CNN architecture, pre-trained on the ImageNet dataset and subsequently trained on various augmented versions of the limited dataset provided; it describes the path to finding an ideal set of parameters and architectures for these models. This paper was very beneficial in offering me an example of how to structure an evaluation research paper. Obviously, my project will involve testing multiple models of classifiers, and evaluating each. This paper helped strategize how to form this research paper, and what structure and format it will take. Please see the table below for an overview of all background research undertaken.

Summary Table of Background Research (See Appendix for more detail):

Year	Title	Results/Accuracy	Year	Title	Results/Accuracy
2018 	Automated plant species identification— Trends and future directions	Deep Learning CNN's are most accurate	2017 	LeafNet: A computer vision system for automatic plant species identification	LeafSnap = 86%, Flavia =95.8%, and Foliage dataset = 97.9%
2018 	UAVs and Machine Learning Revolutionising Invasive Grass and Vegetation Surveys in Remote Arid Lands	96.75% and 96.00% for single mapping of buffel grass and spinifex, respectively, and a multiclass detection rate of 96.54%.	2016 	Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification	Experimental results on the developed model achieved precision between 91% and 98%, for separate class tests, on average 96.3%.
2018 	Plant Classification Using Artificial Neural Networks	Leaf margin and leaf texture have been combined, the algorithm was able to achieve an average accuracy of 97.16%	2016 	Towards Automatic Wild Animal Monitoring: Identification of Animal Species in Camera-trap Images using Very Deep Convolutional Neural Networks	88.9% of accuracy in the Top-1 and 98.1% in the Top-5 in the evaluation set
2017 	Data Augmentation for Plant Classification	Results show that the CNN methods with particular data augmented datasets yield the highest accuracies	2016 	Using Deep Learning for Image-Based Plant Disease Detection	AlexNet=85.53% GoogleNet: 99.34%
2017 	Visual Classifier for Invasive Species		2015 	A Novel Method of Automatic Plant Species Identification Using Sparse Representation of Leaf Tooth Features	BP neural network 73.2±3.3 K-NN 72.3±2.5 Our proposed classifier 76.3±3.4
2017 	Deep Learning for Plant Identification in Natural Environment	ResNet26: 91.78% accuracy	2013 	Automatic classification of legumes using leaf vein image features	87% with the PDA classifier =scanned leaves, 9 % of accuracy for soybean vein characteristics
2017 	Plant species classification using flower images—A comparative study of local feature representations	94% for OF17 dataset	2013 	Leaf based plant identification system for Android using SURF features in combination with Bag of Words model and supervised learning	95.94%
2017 	Deep machine learning provides state-of-the-art performance in image-based plant phenotyping	(>97% accuracy) for root and shoot feature identification and localization	2005 	Mapping invasive plants using hyperspectral imagery and Breiman Cutler classifications (Random Forest)	84% (Khat= 0.56) for the spotted knapweed site (Table 1) and 86% (Khat= 0.62) for the leafy spurge site

3 Dataset and Data Preparation

3.1 Dataset:

A scientifically categorized flora image dataset has been provided by Inland Fisheries Ireland. The dataset has over 112Gb of RGB images. There are 28,000 images across 380 different types of species already categorized into the scientific name of the plant by a botanist in Inland Fisheries Ireland. The data presents flora captured from an average of four different viewpoints, as well as varying backgrounds, proximities and quantities. Each plant species also has several different parts of the plant photographed (flower, leaf, stem, seed). There are also varying types of genus, which is a specific type of plant within a more general species family. Many of our identified species in the dataset have been broken down into genus, or subspecies. A unique characteristic of this very rich dataset is the provision of over 4,000 images of unidentified species, ranging across 150 different categories. It's also important to mention that a non-disclosure agreement has been signed with Inland Fisheries Ireland, prohibiting the unauthorized publishing of any images from this dataset. The categorized data will be used to train a deep learning identification model that will be developed. The model will be used to classify images fed into it and return the species in either real time or near real time.

Originally, it was expected the dataset would also contain GPS metadata about where each plant image was captured. However, that information is not available as the majority of the plant species data has been compiled over 15 years of data collection by the Inland Fisheries Ireland. The equipment used to capture the data images at that time did not have the capabilities of recording GPS data. This reduces the scope of the project, eliminating the proposed feature of species location mapping.



Figure 3.1: Example Images from the dataset²

² (1) mentha aquatica, (2) hypericum pulchrum, (3) achillea_millefolium, (4) aponogeton distachyos, (5) dactylorhiza fuchsii, (6) angelica sylvestris, (7) bellis perennis, (8) lythrum salicaria, (9) mimulus moschatus, (10) crocosmia x crocosmiiflora, (11) butomus umbellatus, (12) barbarea vulgaris, (13) allium triquetrum, (14) caltha palustris, (15) scrophularia auriculata, (16) menyanthes trifoliata, (17) arum maculatum, (18) hypericum androsaemum, (19) eupatorium cannabinum, (20) gunnera tinctoria, (21) digitalis purpurea, (22) lysimachia vulgaris, (23) Ranunculus peltatus penicillatus, (24) epilobium hirsutum, (25) iris pseudachorus, (26) impatiens glandulifera, (27) geranium robertianum, (28) ficaria verna, (29) oenanthe crocata, (30) prunella vulgaris, (31) mimulus guttatus, (32) myosotis scorpioides, (33) persicaria maculosa, (34) nymphaea alba, (35) rumex hydrolapathum, (36) senecio jacobaea x aquaticus, (37) scrophularia nodosa, (38) heracleum mantegazzianum, (39) lotus corniculatus, (40) heracleum sphondylium, (41) hypericum tetrapterum, (42) solanum dulcamara.

3.2 Data Preparation:

The original raw dataset provided by IFI required significant data cleaning. Firstly, any data classes that had either the species or genus missing from the class label, were removed. This was done manually as the folders with incomplete identification were marked distinctly in their folder name. Next, the folder containing unidentified species was removed. Many of the plant species class folders also contained subfolders, which held images of individual instances when the data was gathered, on various days and from multiple botanists. A bash script was written to loop through every plant species class, move the contents of each sub-folder in that class, into the original plant species class folder, and delete all now-empty subfolders. A bash script was then written to cycle through each folder and remove any classes with less than 15 images. This was done in order to limit the imbalanced data classes and maintain a minimum level of image samples for each class when using the images for our model training data. This reduced our dataset to 312 plant species classes.

The data within each plant species class was unstructured (i.e. pictures of individual parts of the plant mixed in together). With the intention of creating a combined classifier, the next task was to sort the contents of each plant species into subcategories: flower, leaf, stem, seed. This was a manual task, sorting through all pictures in each class and distributing the image of each plant component into the relevant sub-category. Once this was done, the bash script was re-run to now remove sub-folders with less than 15 images. This further reduced our dataset to 220 plant species.

3.2.1 Background Removal

The next task was to perform some image pre-processing on the data. Initially, the goal had been to create a combined classifier, with a specific classifier each for the flowers, leaves, stems, and seeds subclasses. A major feature of all previous research conducted in this field is the removal of plant image backgrounds. As referenced in paper [6], even if our goal is plant image recognition in a natural environment, the training data must have the background removed for the model to be able to learn the correct and relevant features. For the purposes of plant image identification in a natural environment, for the training model to learn the correct and most relevant features, all background information had to be removed [10].

The current state of the art software for plant image background removal is PlantCV [21]. PlantCV is an open-source image analysis software package designed for plant phenotyping. It is a collection of modular Python functions, which are reusable units of Python code with defined inputs and outputs. PlantCV functions can be assembled into simple sequential or branching/merging pipelines. A pipeline can be as long or as short as it needs to be, allowing for maximum flexibility for users using different imaging systems and analysing features of seed, shoot, root, or other plant systems. [22]

PlantCV contains a VIS Image Pipeline for background removal of RGB images. VIS stands for Visible Infrared, and the VIS Image Pipeline can process images regardless of what type of VIS camera was used (digital camera, mobile phone camera). A key constraint of this pipeline requires plant material to be a distinctly different colour from the image background. This meant the PlantCV software was effective for the flower data, but not for any other of the plant components. Background removal tests and configurations were tested for numerous data removal pipelines and were trialed with the stem, leaf, and seed image data subclasses, but performance was poor. Because of time constraints and prioritizing the core deliverables of this project, the decision was made to focus solely on flower data.

This reduced our production dataset to 91 classes of flower. Upon further manual quality control checks on the flower data available, this was further filtered down to 31 classes. The issues with the un-usable data ranged across blurred images, extreme lighting, and poorly framed images. Choosing to focus exclusively on these 31 high-quality species classes of flower data, there were several steps in the pipeline of the background removal (Figure 3.2 below).



Figure 3.2: Filter Steps in VIS PlantCV Pipeline

Step 1 was to crop the original image (Figure 3.3). This enabled retention of only the area of interest (the flower itself), being centralized in the picture and reducing the dimensions of the unnecessary data in the wider background (Figure 3.4).



Figure 3.3: Original Image

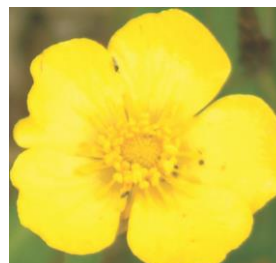


Figure 3.4: Cropped Image

Step 2 in this specific pipeline was to perform pre-masking of the background. The goal is to remove as much background as possible without losing any plant information. In order to perform a binary threshold on an image you need to select one of the colour channels H,S,V,L,A,B,R,G. Here we convert the RGB image to HSV colour space then extract the 'S' or saturation channel, but any channel can be selected based on user need. If some of the plant is missed or not visible, then thresholded channels may be combined (a later step) [23].



Figure 3.5: HSV Filter Image

Step 3 was thresholding the saturation channel. Thresholding involves the creation of a binary image from a grey image based on the specified threshold values. The threshold can be selected to target either the light or dark objects in the image. In this project, the threshold selected was 'light', as the dim green backgrounds are always darker than the actual object target (flower) colour in this dataset.



Figure 3.6: Saturation Thresholded Image

An intermediary step between Step 3 and Step 4 was to apply a median blur filter to reduce noise in the image. It operates by applying a median value to the central pixel within a kernel size. It's important to minimize the use of median blur type steps, as they can cause plant material loss if the blur is too intense. The effect is so subtle it's difficult for the human eye to distinguish, as seen in Figure 3.7 below.



Figure 3.7: Median Blur Filter

Step 4 returns to the original unfiltered cropped image, creating a new branch within the pipeline where a new filter was applied to convert the image from the RGB to the LAB colour space, in which the 'B' or blue-yellow channel is chosen for extraction.



Figure 3.8: LAB Filter Image

Step 5 involved a repeat of Step 3, thresholding the saturation channel, this time on the LAB filtered image. Again, the threshold option selected was 'light'.



Figure 3.9: Saturation Thresholded Image

Step 6 was a combination step, in which the two separate filtered branches in the pipeline, Step 3 (HSV) and Step 5 (LAB), are merged by combining the resulting saturation thresholded images for both. To achieve this, the two images are joined using the bitwise 'OR' operator. The only constraint to performing this operation is that the two image sizes must be identical.

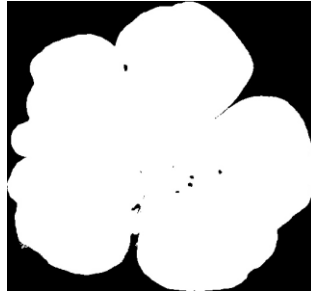


Figure 3.10: Joined Threshold Image

Step 7 was to apply the joined threshold binary image (Step 6) as a mask over the original image (Step 1). This mask functions as a separator, excluding the maximum amount of background without omitting any of the target object plant material.



Figure 3.11: Final Image with Joined Threshold Mask Applied

This pipeline process was repeated for each of our 31 plant classes. Minor variable parameter adjustments were needed for each individual class, in the form of selecting different colour channels to extract (H, S or V for the HSV filter and L, A or B for the LAB filter) depending on the actual colour of the flower.

3.2.2 Data Augmentation:

The next issue that had to be addressed in the pre-processing phase of this project was the number of pictures in each class. Each of the 31 species of flower contained between 15-20 pictures per class. A key component of accurate deep learning models, as outlined in reference [24], is the need for high volumes of training data. To improve overall model accuracy, by increasing the number of pictures in each class, data augmentation was performed. Data augmentation, in machine learning terms, involves predominantly data warping, which is an approach which seeks to directly augment the input data to the model in data space. A very generic and accepted current practice for augmenting image data is to perform geometric and colour augmentations, such as reflecting the image, cropping and translating the image, and changing the colour palette of the image [24].

The augmentation approach used in this research was achieved using an open source computer-vision library called OpenCV. The OpenCV library contains over 500 functions, one of which is the `opencv_createsamples` library. This library was used to create 2 different variations of augmented datasets, both will be outlined below.

These two versions of data augmentation were undertaken in order to test the speed and accuracy of training the model with random image backgrounds (Version 1 of augmentation), which would be consistent with our desired project use-case of photographing a plant in its natural environment and instantly being provided a classification prediction for that plant class. This is compared with the speed and accuracy of training a model with all backgrounds removed / black backgrounds (Version 2 of augmentation), which would involve our use-case now including the pre-processing of the user-input image to remove the background before classifying and returning them a result. The performance results of training using both these augmentation techniques are outlined and compared in later sections, during the model evaluations.

The `opencv_createsamples` library operates on the premise of “positive” and “negative” image datasets. We consider a positive image (Figure 3.12) to be a picture containing the target object, which in this case is a flower. For this project, the positive image dataset is the data we currently already have (i.e. the 31 classes of plants with backgrounds removed).



Figure 3.12: Example of a 'positive' image

A negative image (Figure 3.13), in the context of this library, is an image that does not contain the target object (i.e. an image not containing the flower). For the first version data augmentation in this project, the first step was creating a supplementary dataset of 600 simple abstract pictures (none of which contained any of our existing classes of flower). The size of 600 was selected in order to minimise repetition of backgrounds used and allows a large enough pool for random selection from the dataset to be viable.



Figure 3.13: Examples of 'negative' images

The `opencv_createsamples` library operates by individually taking 1 of the 15-20 positive pictures in a particular plant class, and performing transformations on the actual flower object, before superimposing it onto a negative or ‘random’ image (Figure 3.14). This process of transforming and superimposing onto a negative background image is then repeated for a user-specified number of times for that particular positive image. This is then repeated for every image, in every plant class. For the purpose of minimizing bias, the classes were balanced by tailoring the number of augmented images needed for each particular class, based on creating the necessary amount to supplement the data in each class to a total of 400 positive images per class.



Figure 3.14: Examples of Version 1 augmented

The second version of data augmentation tested for this project involved the same process as above, but with key difference of the “negative” image dataset consisting exclusively of plain black images instead of random backgrounds (Figure 3.15). Both versions of the augmented dataset were tested and evaluated when training predictive models in a later section of this report.

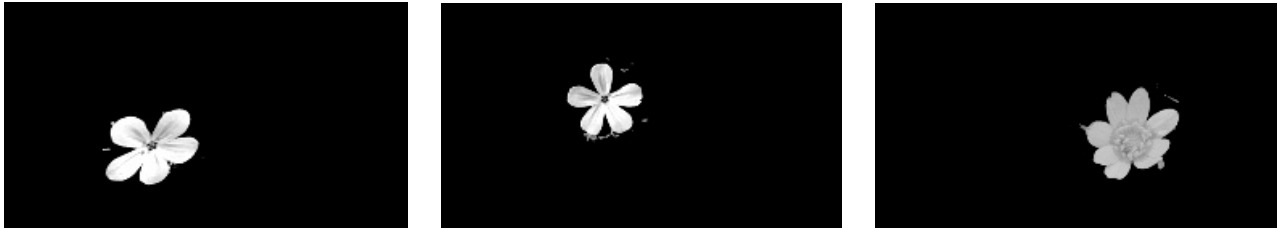


Figure 3.15: Examples of Version 2 augmented

4 Software Development and Analysis:

Once the dataset preparation was completed, the system implementation took place in order to create the training model to classify our plant species images. The software development for this project can be broken down into two individual software components. The first is the software to build and train a deep learning convolutional neural network classifier using the training data pre-processed in the previous section. The second component is an online classification system, that will enable the user to interact with the actual classifier. The overall software system architecture can be seen in Figure 4.1 below.

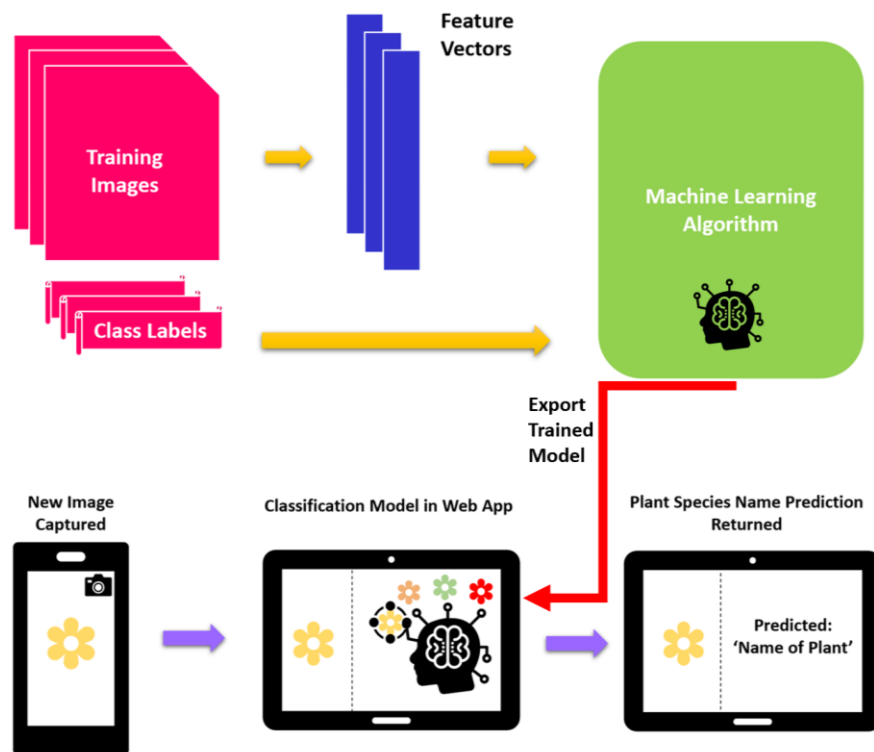


Figure 4.1: Software Design Component Architecture

The first component developed, and the core priority task for the project overall, was the machine learning algorithm. Software was written to handle the entire pipeline process: loading the training data images, applying machine learning techniques, before finally creating a final machine learning algorithm that is highly accurate. As mentioned previously, a CNN classifier was chosen as the model for development. To briefly focus more on that architecture, and actual software deliverable extracted from that process, see the below diagram, Figure 4.2.

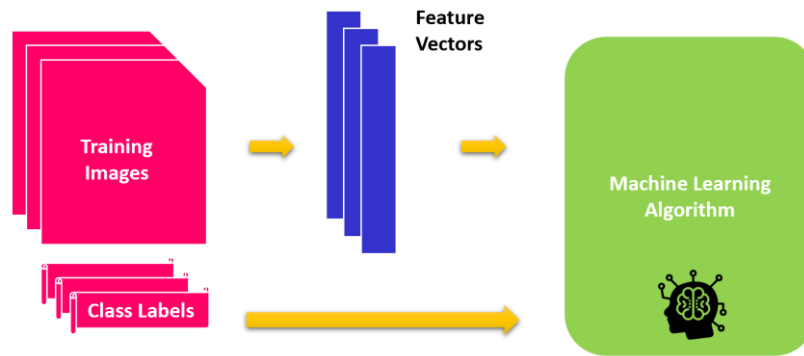


Figure 4.2: CNN Process Software Design

Without diving too much into the details of this CNN implementation, the actual software that was written using libraries from a deep learning framework, was simply a means of training and testing the model. The data was split at the start of the process, 70% training data, 30% testing data. This will enable accurate evaluations of ‘unseen’ data (i.e. our use-case). Code was written to load in all the training data and labels, split it, automatically extract feature vectors, before being passed to the CNN model layers, and trained to create a machine learning algorithm that can classify plant images. Evaluation code was written to test the classification accuracy of the algorithms. Once a high accuracy of classification performance was achieved, the trained machine learning algorithm is exported as a plant classification model with stored feature weights. An overview of that overall software component network architecture is outlined in Figure 4.3. The training engine refers to the software written using the Tensorflow and Keras libraries in order to create CNN models. Each variation of model generated was evaluated based on classification accuracy scores. The most accurately trained model was then exported in a .h file format (standard format for multidimensional arrays of scientific data). That classifier model then formed the backend to our second software component- an online identification system.

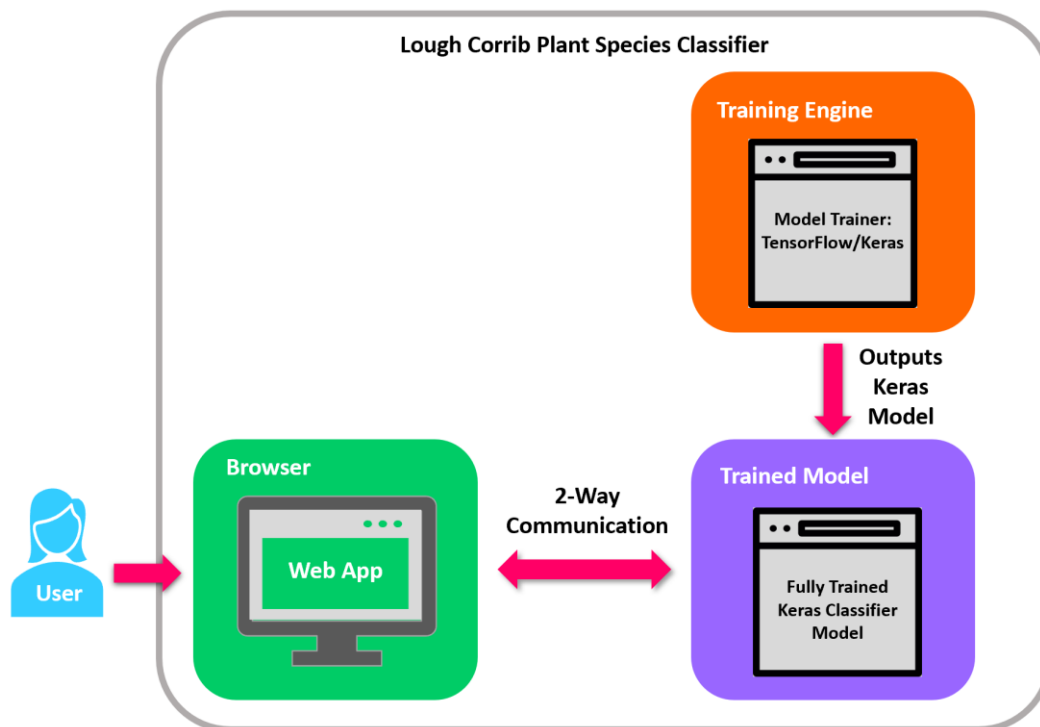


Figure 4.3: Software Components Network Architecture

The online plant identification system, shown here on the bottom left of the diagram, is the exclusive method of user interaction with the model. The web app essentially acts as a front end interface for the user to utilize the trained plant classifier model. The user can access the web app through a browser, and

the web page itself is complete with two-way communications for interacting with the trained model in the backend. This enables the user to upload a new image to the web app, the web app exports that image to the trained model, the model classifies the image, and returns a plant class prediction which is then displayed on the web app for the user.

Figure 4.4 below further offers an even higher-level overview of the software system use case in this project. A new image is captured, it is uploaded to the classification model, and the classification model then returns its prediction for that new image. That is the core functionality of classification this project must achieve. In the simplest terms possible, that is what our target deliverable is.

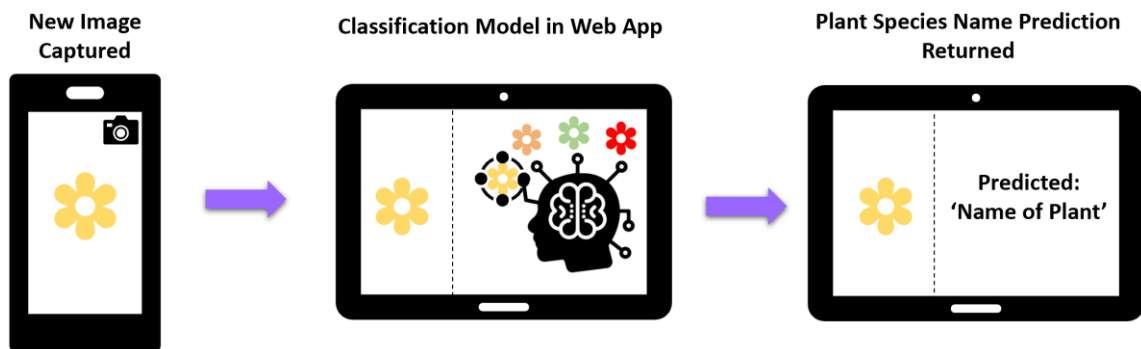


Figure 4.4: Software System Core Use Case

Overall, we can conclude that the current state of the art approach to supervised machine learning for image plant species identification, based on the concluding findings of the research summarised in [1], is the supervised CNN model architecture outlined above. That model, and the online classification web app are both key deliverables in this overall software system.

5 Detailed Design and Software Implementation:

Following on from the high-level software architecture diagrams described above, the next task was to start implementing these designs by actually coding software. The first phase of implementation undertaken was developing the deep learning classification model.

5.1 CNN

There are many different frameworks widely available for deep learning. The chosen framework used for this project is TensorFlow. TensorFlow is a machine learning system that operates at large scale and in a heterogeneous environment [25]. TensorFlow uses dataflow graphs to represent computation, shared state, and the operations that mutate that state. It maps the nodes of a dataflow graph across many machines in a cluster, and within a machine across multiple computational devices, including multicore CPUs, general-purpose GPUs, and custom-designed ASICs known as Tensor Processing Units (TPUs). It enables experimentation with novel optimizations and training algorithms. TensorFlow supports a variety of applications, with a focus on training and inference on deep neural networks. It is this purpose built focus on deep CNN's that makes the TensorFlow framework perfect for implementing this project.

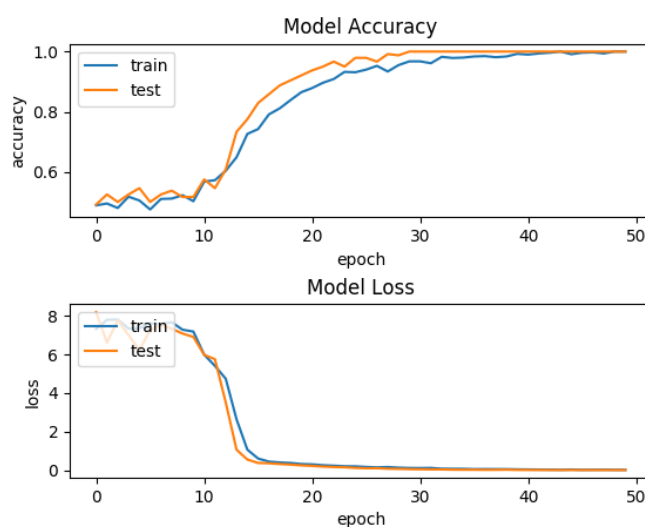
As mentioned in a previous chapter on Data Pre-processing, there are 2 versions of augmented data to test in the models. Version 1 is data augmented with random backgrounds. Version 2 is the data augmented on just plain black backgrounds. The first step in this implementation is to create a simple CNN model and

compare how the model performs on both versions of data, before choosing to move forward with the most suitable version of data for this project.

As emphasised in the previous section on Background Research for the current state of the art technology in this field, the overall deep learning model architecture is based on Figure 2.1. Building on that template, the first model implemented is a CNN with 1 layer. Starting with the standard TensorFlow CNN architecture and coding the implementation in python, the model begins by importing the TensorFlow and related libraries. The next step is the load data function, this function reads all the image data and their corresponding species-name as images and label arrays. These arrays are then divided into training_data and test_data sub-arrays which are loaded into the model. The model itself is a very simple iteration, consisting of just one fully connected layer, an accuracy and loss metrics calculator and a training optimizer. The model will train using the training_data (images, labels) arrays and validate using the test_data arrays. Both versions of augmented data (random background and black background) were used to train models with this architecture. A visual summary of their performance results is displayed below in Figure 5.1 and Figure 5.2.



**Figure 5.1: Data Augmentation:
Version 1 - Random Background**



**Figure 5.2: Data Augmentation:
Version 2 - Black Background**

However, these train and test accuracy results don't show the whole story (see the Evaluation Section below for a detailed analysis of performance accuracies). While the model trained with version 2 of augmented data (black background) has a higher accuracy score at training time, and it is more accurate at identifying new images as long as the background is also removed, performance greatly deteriorates when this model is used to identify new plant images in their natural background (the normal project use-case of a plant image captured in its natural habitat with green foliage in the background.) In contrast, the model trained with version 1 of augmented data (random picture backgrounds), while having lower accuracy at training time, it performed significantly better on plant images in a natural background. It's identification accuracy of new unseen images averages approximately the same as its accuracy scores on the random background test data.

Therefore, an important design decision had to be made. Option 1 was prioritizing the higher classification accuracy by adding image processing pipeline features to the online identification system, where each new user image would have the background removal process applied prior to classification by the model trained with black backgrounds (version 2 of data augmentation). Option 2 was prioritizing user experience, specifically the speed of instant classification by the model trained with version 1 of the data augmented with random backgrounds (version 1 of data augmentation) without needing to spend time pre-processing and removing the background before every classification prediction is returned.

Both design options were considered equally. After much strategic analysis, Option 2 was decided as the most feasible implementation moving forward with the project. While the higher accuracies of removing the background each time a new image is classified was a strongly desired solution, ultimately the time constraints of this project eliminated that possibility. As outlined in the Data Pre-processing chapter, particularly when describing the use of the PlantCV VIS background removal pipeline, the big obstacle here was the need to customize the pipeline variables to various configurations depending on the colour and lighting of that particular plant. This need to manually adjust the pipeline variables depending on the flora colours, meant building a highly complex background removal pipeline, with several different deployment configurations specific to every colour of plant. Unfortunately, that type of substantial pre-processing pipeline was outside the scope of this project. On the basis of that, the more feasible Option 2 was pursued, using the data augmented with random background images to train all remaining models evaluated in this project.

The next phase was to increase both the number of plant classes and the overall classification accuracy. This was achieved by adding more layers and creating a deeper model. To move incrementally and preserve a scientific process, layers were added individually. After adding 16 plant classes and another layer to the TensorFlow framework, it became stagnant before dropping accuracy. With 16 classes, it was no longer performing above 10% accuracy in classification. Keras was then introduced. Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow. See the table below (Figure 5.3) for a comprehensive general structure outline of all models built and tested for this project.

Model	Framework	Structure	Data Augmentation Version	Number of Classes
Model_1	TensorFlow	1 FC layer	Version 1	2
Model_2	TensorFlow	1 FC layer	Version 2	2
Model_3	TensorFlow	1 FC layer	Version 1	10
Model_4	TensorFlow	1 FC layer	Version 1	16
Model_5	Keras	2 conv layers, 2 FC layers	Version 1	2
Model_6	Keras	2 conv layers, 2 FC layers	Version 1	5
Model_7	Keras	2 conv layers, 2 FC layers	Version 1	15
Model_8	Keras	3 conv layers, 2 FC layers	Version 1	2
Model_9	Keras	3 conv layers, 2 FC layers	Version 1	5
Model_10	Keras	3 conv layers, 2 FC layers	Version 1	13
Model_11	Keras	3 conv layers, 2 FC layers	Version 1	31

Figure 5.3: Summary Table of Deep Learning Model Structures Implemented

Iterating through deeper Keras models in the format structure outlined in the table above, eventually the final and most successful model is a 5-layer Keras CNN (Model_11). See the detailed architecture diagram below. The input layer takes a 64 x 64 pixel flora image from the pre-processed plant species dataset. The first layer has a hidden convolutional and MaxPooling operation. This layer has a filter size of 40. Each filter runs across the input image and creates a 2-D feature activation map. This activation map takes the form of

a binary vector. The MaxPooling operation, as described in the section above on 'The Pooling Layer', is then performed to reduce the maximum output within a rectangular neighbourhood. This pools the common parameters, helping eliminate noise in the image and instead focus on key feature identification. It helps combat overfitting. This output is then fed forward sequentially to the next layer. Our second layer is another convolutional and MaxPooling operations layer. In order to improve accuracy, this layer has an increased filter size of 80. After MaxPooling is applied within that layer, the output is again forwarded to our next layer. Similar to the previous two layers, the third layer in our model is also comprised of a convolutional and MaxPooling layer. Again, in order to incrementally increase accuracy, the filter size is increased to 164. The output of that layer is then fed to an intermediary layer, called a flatten layer. This flatten operation takes the output dimensions from the convolutional layers and reshapes to match the size of the original input layer dimensions, 64x64. This is done in order to connect it to the next layer - a fully connected layer, or FCL. As mentioned above, an FCL connects every neuron in one layer, to every layer in another. While the layer four FCL is size 64x64 in correspondence with the input layer, it is passed onto our fifth and final layer, another FCL. The size of our second FCL is 31, each node corresponding to one of our 31 plant species classes in the dataset. This final FCL is then connected to our concluding output layer, which returns the predicted classification label for the input image.

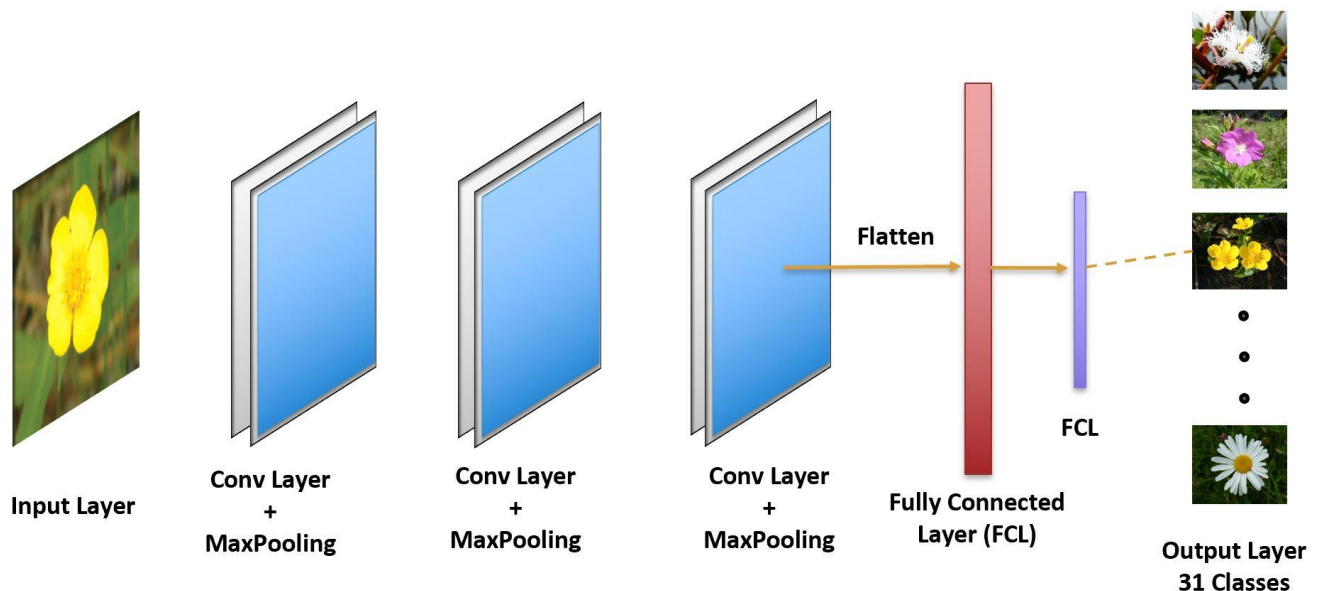
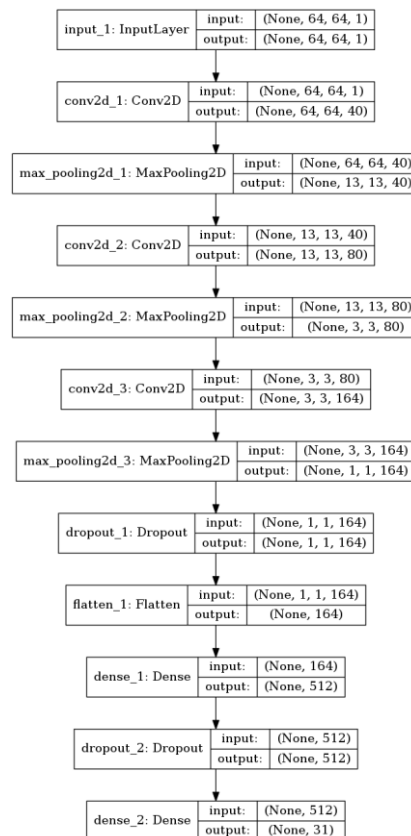


Figure 5.4: Keras 5 Layer CNN Architecture for final Model

A more in-depth and exhaustive outline of the model structure and specific layer configurations is outlined below in Figure 5.5. As mentioned already, we start with our image input size 64x64 pixels. The first layer is Convolutional layer 1 that takes the 64x64 input and passes it through a filter size of 40. This output is then passed onto MaxPooling that reduces the size to 13x13 pixels. Convolutional layer 2 takes an input size of 13x13, applies a filter of size 80. The next MaxPooling layer takes that input of 13x13 and reduces the output size to 3x3. Convolutional layer 3 takes that input and applies a filter of 164. The third MaxPooling operation takes the input of 3x3 and reduces it to an output size 1x1. This is passed to a dropout layer, in which randomly selected neurons are ignored in training in order to prevent overfitting. This operation does not affect the output size. The flatten layer takes the output dimensions 1x1 from the previous layer and reshapes it to match the size of the original input layer dimensions 64x64. Next, a fully connected layer(dense layer) takes input 64x64 and connects to the next layer. An additional dropout layer is added next for further protection against overfitting. Finally, every neuron from Fully Connected Layer 1 is connected to every neuron in Fully Connected layer 2. In line with having 31 plant classes in our model, Fully Connected layer 2 has 31 neurons corresponding to a flora species, which form the predicted classes in the output layer.



**Figure 5.5: Final Keras 5 Layer CNN
Technical Model Structure Specification**

5.2 Online Classification System App

The next step, once the model has been trained on the above architecture, is to save it and export it in a .h5 file format. This leads to the concluding step in this overall implementation - creating an online system whereby plant images can be uploaded and classified by a user. To limit the scope of this feature and prioritize fast prototyping and delivering an effective and functional final project, a web application was developed in favour of a mobile app. The architecture of the App can be found below (Figure 5.6)

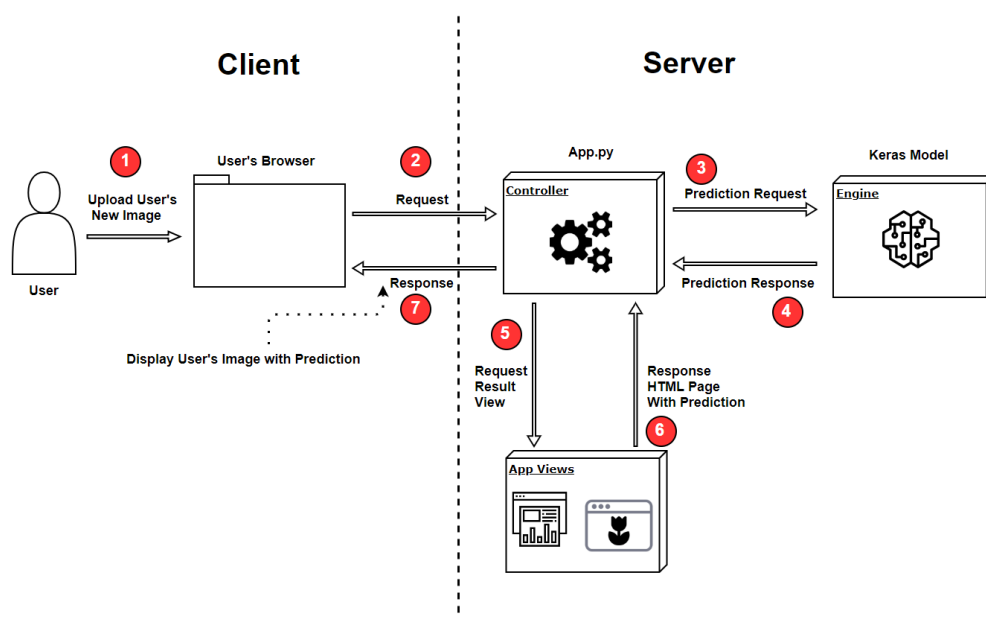


Figure 5.6: Online Classification System Implementation Architecture

The web app was developed using Flask. Flask is a micro web framework written in python. It is highly compatible with other frameworks and allowed for the easy importing of the Keras model. For the design styling of the app, a Bootstrap template was used to give a professional look. Bootstrap is an open source toolkit for developing with HTML, CSS, and JS [26].

The online classification system demonstrated in Figure 5.6 above works as follows: at start time the script (App.py) initiates a CNN object to which the Keras model is loaded into. When the user submits the image to be classified (Step 1) it passes that image through the app controller (Step 2) to the model's prediction function (Step 3). The Keras model returns the prediction response (Step 4) to the app controller. The app then requests the 'Results View' from the app views (Step 5), which returns a response of an HTML page with the plant species name prediction (Step 6). This is then returned through the app in the browser, which displays the user image and the predicted plant class (Step 7). Finally, it clears the prediction function of the model, ready for reuse.

The app user experience was designed to be both quick and easy to use. To upload a new image to classify, you click the 'Browse...' button in the centre of the home screen. That opens your device file system, allowing you to select an image. Once your selected image is loaded to the web page, you can click 'Submit' and your image will be instantly classified. The app architecture itself consists of 2 pages: a home page and a test page. The home page is displayed below in Figure 5.7, outlining the actual dashboard for interacting with the model. The test page contains the model performance accuracy graphs, as shown later in this section.

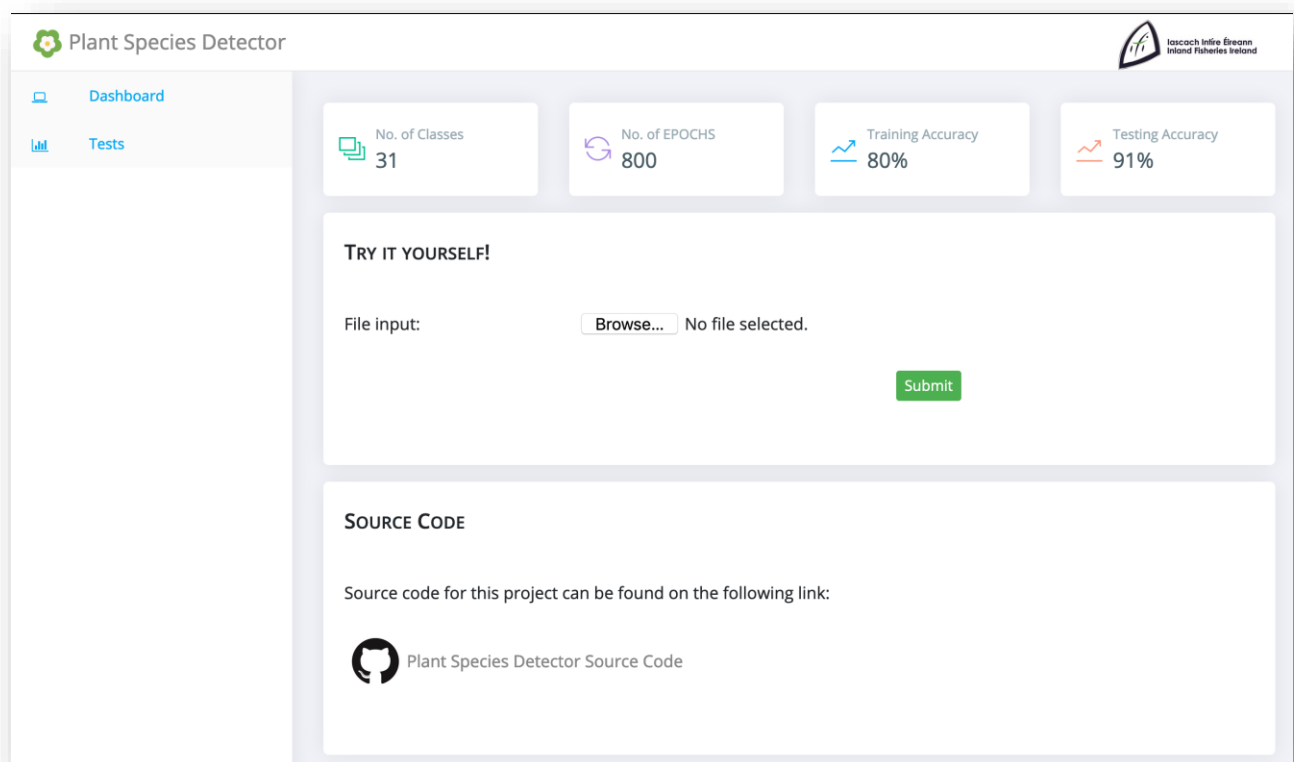


Figure 5.7: Screenshot of Web App Home Page Dashboard

The plant species which has the highest probability score is returned as the predicted class. The user's selected image is displayed on the screen, as well as the predicted plant class label and model's percentage confidence in the prediction (Figure 5.8 & 5.9). However, should the predicted accuracy be below the reasonable threshold of 60% at which we deem prediction confidence is not high enough, the webpage is programmed to inform you that the species captured in the uploaded image does not yet exist in the database. To classify a new image, you simply re-click 'Browse...' and repeat the steps.

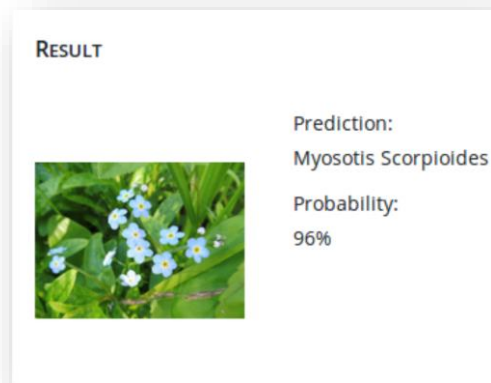
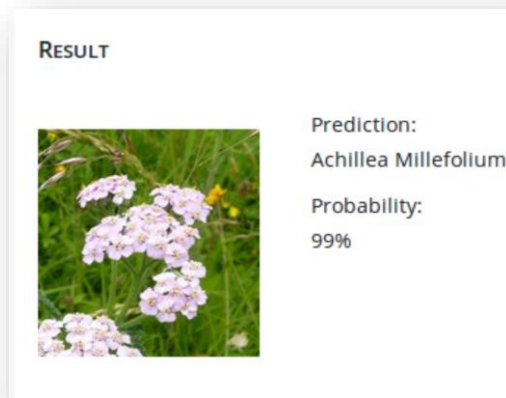


Figure 5.8 and Figure 5.9: Examples of Images Classified by the Online Classification System Web App

6 Testing/Evaluation

The main method of evaluation in this project is the classification accuracy of the models implemented. An exhaustive overview of the entire collection of models evaluated during the implementation of this project is summarised in Figure 6.6 at the end of this section below.

The first element of implementation to evaluate is the type of augmented data used. As was detailed earlier in the Data Pre-processing Chapter, there were 2 versions of augmented data used. The first had random backgrounds (Figure 6.1) and the other had plain black backgrounds (Figure 6.2).



Figure 6.1: Data Augmentation: Version 1 - Random Background

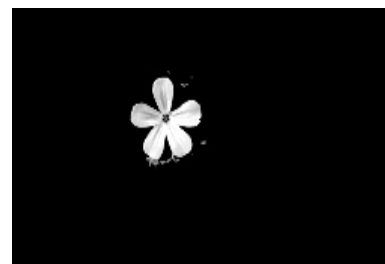


Figure 6.2: Data Augmentation: Version 2 - Black Background

Both of these versions of augmented data were tested separately on a single layer TensorFlow CNN. A visual display of the results is demonstrated below.

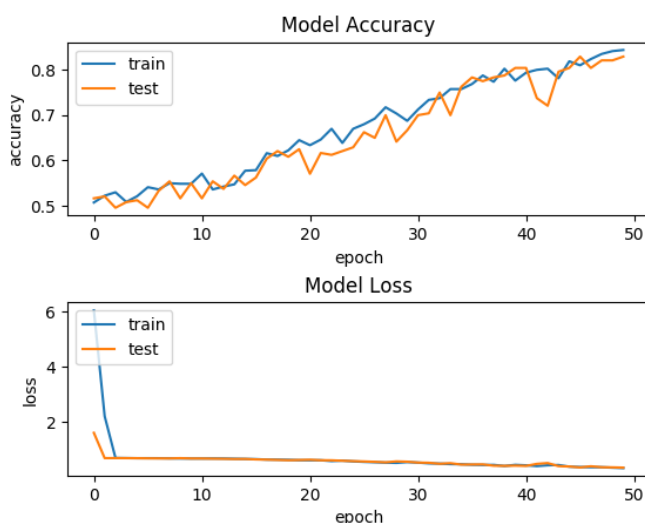


Figure 6.3: Accuracy of Data Augmentation: Version 1 - Random Background

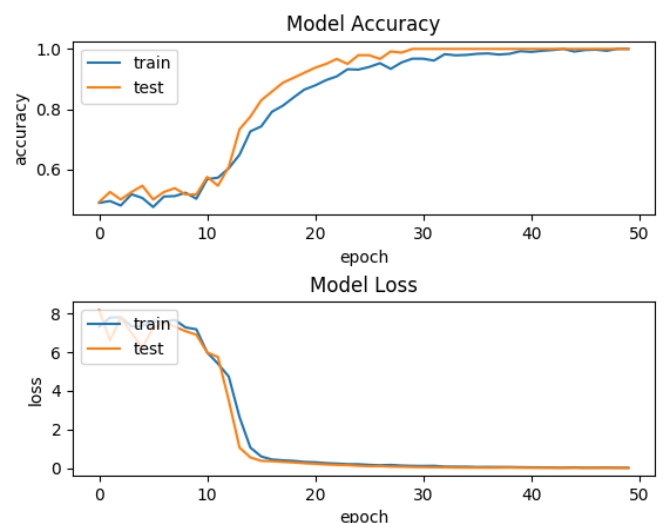


Figure 6.4: Accuracy of Data Augmentation: Version 2 - Black Background

To reiterate the findings outlined earlier in this report, while the model (Fig 6.6, Model_2) trained with the version 2 of augmented data (black background) has a higher testing accuracy score (100%) at training time, and it is more accurate at identifying new images as long as the background is also removed, performance accuracy drops (<65%) when this model is used to identify new plant images in their natural background (the normal project use-case of a plant image captured in its natural habitat with green foliage in the background.)

In contrast, the model (Fig 6.6, Model_1) trained with version 1 of augmented data (random picture backgrounds), while having lower accuracy (93%) at training time, it performed significantly better on plant images in a natural background. It's identification accuracy of new unseen images averages approximately the same as its accuracy scores on the random background test data (>85%).

Based on the justification provided in the earlier section on Implementation, version 1 (data augmented with random backgrounds) was then used to train and test all subsequent models in this project. The strategy adopted in building these classification models was an iterative process. Starting with just a single fully connected layer model, it was trained on 10 classes of plant species (Fig 6.6, Model_3), with an increase in epochs to help improve accuracy (65%). The single FC TensorFlow CNN was exhausted with a maximum of 16 plant classes (Fig 6.6, Model_4) before accuracy plummeted (to less than 10%).

Next, again as outlined in the previous section, Keras was chosen as the framework for building a deeper CNN. Starting with just 2 Convolutional layers and 2 Fully Connected layers, a model (Fig 6.6, Model_5) was trained on 2 plant classes, yielding 86% accuracy. While that was lower than our previous initial accuracy scores for Model_1, as classes were added, performance reduced to 80% accuracy for 5 classes (Fig 6.6, Model_6) and 77% accuracy for 15 classes (Fig 6.6, Model_7). Increasing epochs no longer improved performance for these models, leading to the next step, adding another layer. A third convolution layer was added, meaning the model now comprised of 5 layers in total (3 Conv layers, 2 FC layers.) This new layer model architecture was tested again, initially on 2 plant classes (Fig 6.6, Model_8). Accuracy was found to be fantastic for this model architecture (99% accuracy), so the iterative process of adding more plant classes began. It started with a small increase, up to 5 plant species classes (Fig 6.6, Model_9), which yielded an accuracy of 96%. Continuing to add more classes, a model (Fig 6.6, Model_10) was trained on 9 classes, and yielded an accuracy of 93%.

Finally, the most accurate model created in this project (Fig 6, Model_11) is a 5-layer Keras CNN. It contains 3 Conv layers and 2 FC layers. This model, trained on 12,400 images (400 images per flora class), to classify all 31 types of plant species in our dataset, achieves an accuracy score of 91%. Experiments were run, adding additional layers, re-ordering the existing layer configurations, but accuracy did not improve. Various hyper-parameter tuning techniques such as varying the number of epochs ran, as well as testing a range of filter sizes for the Conv layers, ultimately led to no significant increase in accuracy.

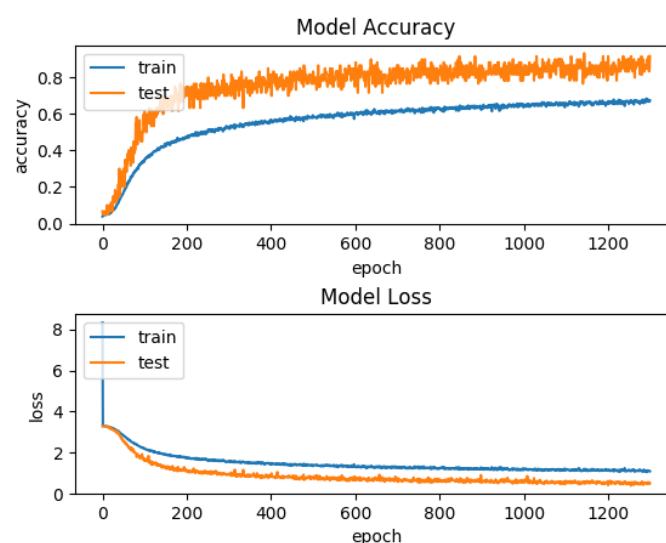


Figure 6.5: Evaluation Accuracy of 5 layer Keras CNN with 31 classes

Therefore, Model_11, with 5 layers and 91% testing accuracy is the most accurate model for this project, and thus was the model used to create the online web app classification system. This implementation is subsequently delivering on the core objective of this project, as outlined in the specification 'Create a system whereby flora can be identified by mobile imagery with a high degree of accuracy using deep learning models.'

Model	Framework	Structure	Data Augmentation Version	Number of Classes	Epochs	Testing Accuracy at Training
Model_1	<i>TensorFlow</i>	<i>1 FC layer</i>	<i>Version 1</i>	<i>2</i>	<i>50</i>	<i>93%</i>
Model_2	<i>TensorFlow</i>	<i>1 FC layer</i>	<i>Version 2</i>	<i>2</i>	<i>50</i>	<i>100%</i>
Model_3	<i>TensorFlow</i>	<i>1 FC layer</i>	<i>Version 1</i>	<i>10</i>	<i>120</i>	<i>65%</i>
Model_4	<i>TensorFlow</i>	<i>1 FC layer</i>	<i>Version 1</i>	<i>16</i>	<i>300</i>	<i><10%</i>
Model_5	<i>Keras</i>	<i>2 conv layers, 2 FC layers</i>	<i>Version 1</i>	<i>2</i>	<i>70</i>	<i>86%</i>
Model_6	<i>Keras</i>	<i>2 conv layers, 2 FC layers</i>	<i>Version 1</i>	<i>5</i>	<i>90</i>	<i>80%</i>
Model_7	<i>Keras</i>	<i>2 conv layers, 2 FC layers</i>	<i>Version 1</i>	<i>15</i>	<i>180</i>	<i>77%</i>
Model_8	<i>Keras</i>	<i>3 conv layers, 2 FC layers</i>	<i>Version 1</i>	<i>2</i>	<i>175</i>	<i>99%</i>
Model_9	<i>Keras</i>	<i>3 conv layers, 2 FC layers</i>	<i>Version 1</i>	<i>5</i>	<i>175</i>	<i>96%</i>
Model_10	<i>Keras</i>	<i>3 conv layers, 2 FC layers</i>	<i>Version 1</i>	<i>13</i>	<i>220</i>	<i>93%</i>
Model_11	<i>Keras</i>	<i>3 conv layers, 2 FC layers</i>	<i>Version 1</i>	<i>31</i>	<i>1500</i>	<i>91%</i>

7 Conclusions and Future Work

7.1 Future Work

There are some immediate feature extensions that would advance the quality of this project. First and foremost, the most intuitive next step would be to add more species of flower plant classes that can be classified. While the model is capable of identifying the 31 classes of usable data provided by the IFI dataset, it would be beneficial to source additional quality training data and continue adding more species of plant classes. Furthermore, while this project was focused solely on flower plant class identification due to the data available, it would be interesting to develop a combined classifier, that could identify various components of the plant (flower, leaf, stem) and calculate the predicted class based on the overall plant anatomy combination of features, not just the flower. This would help make the classifier model more robust against current limitations such as the flowers components of plants dying in winter, preventing potential identification.

Another immediate feature I would like to add, as was originally included in this project scope, the aim to further enhance understanding of the Flora composition around Lough Corrib by creating a mapping of species using GPS metadata. Unfortunately, as mentioned earlier in the report, the GPS metadata was unavailable for this plant species image dataset. As a future development to this current project, delivering a species GPS documentation and mapping function would be highly useful to geographically monitor flora populations around Lough Corrib. Invaluable insights could be gained through GPS data associated with flora images for global map visualisations and monitoring invasive species spread (i.e. *Lagarosiphon* invasive species in Lough Corrib).

On a broader note, there are many exciting areas emerging in the field technology for plant species. While many current studies operate on relatively small and non-representative datasets such as the one used in this project, only a select few studies train CNN plant species classifiers on much larger flora image datasets. The most recent research trends are now gravitating towards applications of transfer learning. This would mean pre-training a classifier on a large dataset, such as ResNet, before actual training begins. The classifier will then only be fine-tuned to the specific plant species classification problem by the training of a small number of high-level network layers proportional to the amount of available problem-specific training data [1]. Considering the limited size of this dataset, investigating the application of transfer learning with the aim to improve accuracies would be an interesting development in the project.

Over a decade ago, the authors of [27] proposed that successful plant species identification developments would rely on interdisciplinary research collaborations between both biologists and computer scientists. The majority of automated plant species identification today is undertaken in academic research surrounding computer vision and machine learning. Research should move towards more interdisciplinary endeavours. Biologists can apply machine learning methods more effectively with the help of computer scientists, and the latter is able to gain the required exhaustive understanding of the problem they are tackling by working with the former [1].

In terms of taking a closer look at the topic of plant phenotyping, all approaches for improving crops require measurement of traits (phenotyping) [28]. Adapting the same methodology used in this project and applying it to plant's image classification in identifying features for stress phenotyping that impacts specific plants with a high economic impact for Ireland would be a highly impactful application of this research. Machine learning approaches enable the ability to learn patterns from data, while automatizing decisions in terms of plant's stress without programming explicit rules. As an ultimate future vision of this project, drone-collected data could provide a valuable source of information for flora on earth with further prospects of exploitation for ecological monitoring and biodiversity conservation as well as monitoring the effects of climate change.

7.2 Conclusion:

The core aim of this project was fulfilled successfully, through the creation of a system whereby flora can be identified by plant imagery with a high degree of accuracy using deep learning models.

Based on the evaluation analysis section above, the highest performing model was the Convolutional Neural Network with 5 layers. This 91% accuracy satisfies the core objective of this project, to classify plant species images in Lough Corrib 'with a high degree of accuracy using deep learning models.' Additional project objectives were also achieved, particularly in following scientific process in 'selecting and testing a number of deep learning models and training them with the image dataset.' Eleven different Deep Learning models were trained and evaluated to document 'the speed and robustness of the flora identification using the selected deep learning models'. The biggest unforeseen challenge in this project was the sheer amount of pre-processing and data cleaning needed to create a usable high-quality production dataset. For the advanced tasks 'the initial focus is on the development of a classification algorithm based on the given image dataset and the calculation of the accuracy' the biggest challenge proved to be maintaining high accuracy scores when increasing the number of flora classes to be identified. The more flower classes added, the less accurate the model became. It was important to achieve a model algorithm that could offer both a high quantity of classification categories (31 flowers), as well as preserving high classification accuracies across all those flora categories. In the end, this was achieved through significant data pre-processing and adding additional layers to the network depth. An online web page classification system was also successfully implemented for easy use by Inland Fisheries Ireland botanists. This satisfied the concluding objective of this project scope 'creating an online system to allow for documentation to be uploaded and classified'. This system will subsequently be deployed to the botanists at Inland Fisheries Ireland, enabling them to utilize this project as a plant classification tool.

8 References:

- [1] Wäldchen, J., Rzanny, M., Seeland, M., & Mäder, P. (2018). Automated plant species identification—Trends and future directions. *PLOS Computational Biology*, 14(4). doi: <https://doi.org/10.1371/journal.pcbi.1005993>
- [2] Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding Machine learning: From Theory to Algorithms. Retrieved from <http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning>
- [3] Goodfellow, I., Bengio, Y., & Courville, A. (2016): Deep Learning. Retrieved from <http://www.deeplearningbook.org/>
- [4] Wung, J., & Perez, L. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. doi: <http://arxiv.org/abs/1712.04621>
- [5] Haykin, S. (2009). Neural Networks and Learning Machines. Retrieved from <http://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf>
- [6] Sun, Y., Liu, Y., Wang, G., & Zhang, H. (2017). Deep Learning for Plant Identification in Natural Environment. *Computational Intelligence And Neuroscience*, 2017, doi: <https://doi.org/10.1155/2017/7361042>
- [7] Barré, P., Stöver, B., Müller, K., & Steinhage, V. (2017). LeafNet: A computer vision system for automatic plant species identification. *Ecological Informatics*, 40, 50-56. doi: <https://doi.org/10.1016/j.ecoinf.2017.05.005>
- [8] R., Craviotto, R., Arango, M., Gallo, C., & Granitto, P. (2014). Automatic classification of legumes using leaf vein image features. *Pattern Recognition*, 47(1), 158-168. doi: <https://doi.org/10.1016/j.patcog.2013.06.012>
- [9] Mohanty, S., Hughes, D., & Salathé, M. (2016). Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers In Plant Science*, 7. doi: <https://doi.org/10.3389/fpls.2016.01419>
- [10] Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification. *Computational Intelligence And Neuroscience*, 2016, 1-11. doi: <https://doi.org/10.1155/2016/3289801>
- [11] Nguyen, Q., Le, T., & Pham, N. (2013). Leaf based plant identification system for Android using SURF features in combination with Bag of Words model and supervised learning. doi: <https://doi.org/10.1109/ATC.2013.6698145>
- [12] Jin, T., Hou, X., Li, P., & Zhou, F. (2015). A Novel Method of Automatic Plant Species Identification Using Sparse Representation of Leaf Tooth Features. *PLOS ONE*, 10(10), e0139482. doi: <https://doi.org/10.1371/journal.pone.0139482>
- [13] Pawara, P., Okafor, E., Schomaker, L., & Wiering, M. (2017). Data Augmentation for Plant Classification. *Advanced Concepts For Intelligent Vision Systems: 18Th International Conference*. doi: https://doi.org/10.1007/978-3-319-70353-4_52
- [14] Seeland, M., Rzanny, M., Alaqraa, N., Wäldchen, J., & Mäder, P. (2017). Plant species classification using flower images—A comparative study of local feature representations. *PLOS ONE*, 12(2), e0170629. doi: <https://doi.org/10.1371/journal.pone.0170629>




- [15] Pound, M., Atkinson, J., Townsend, A., Wilson, M., Griffiths, M., & Jackson, A. et al. (2017). Deep machine learning provides state-of-the-art performance in image-based plant phenotyping. *Gigascience*, 6(10). doi: <https://doi.org/10.1093/gigascience/gix083>
- [16] Pacifico, L., Macario, V., & Oliveira, J. (2018). Plant Classification Using Artificial Neural Networks. doi: <http://dx.doi.org/10.1109/IJCNN.2018.8489701>
- [17] Gomez Villa, A., Salazar, A., & Vargas, F. (2017). Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks. *Ecological Informatics*, 41, doi: <https://doi.org/10.1016/j.ecoinf.2017.07.004>
- [18] Lawrence, R., Wood, S., & Sheley, R. (2006). Mapping invasive plants using hyperspectral imagery and Breiman Cutler classifications (randomForest). *Remote Sensing Of Environment*, 100(3), 356-362. doi: <https://doi.org/10.1016/j.rse.2005.10.014>
- [19] Mengersenand, K., Gaston, K., Gonzalez, F., & Sandino, J. (2018). UAVs and Machine Learning Revolutionising Invasive Grass and Vegetation Surveys in Remote Arid Lands. *Sensors*, 18(2), 605. doi: <https://doi.org/10.3390/s18020605>
- [20] Jobson, E., & Hernandez, A. (2017). Visual Classifier for Invasive Plant Species. Retrieved from <http://cs231n.stanford.edu/reports/2017/pdfs/944.pdf>
- [21] PlantCV. (2019). Retrieved from <https://plantcv.danforthcenter.org/>
- [22] Gehan, M., Fahlgren, N., Abbasi, A., Berry, J., Callen, S., & Chavez, L. et al. (2017). PlantCV v2: Image analysis software for high-throughput plant phenotyping. *Peerj*, 5, e4088. doi: <https://doi.org/10.7717/peerj.4088>
- [23] Team, PlantCV. (2019). VIS pipeline - PlantCV. Retrieved from https://plantcv.readthedocs.io/en/latest/vis_tutorial/
- [24] Wung, J., & Perez, L. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. doi: <http://arxiv.org/abs/1712.04621>
- [25] Abadi, M. et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Retrieved from <http://download.tensorflow.org/paper/whitepaper2015.pdf>
- [26] Mark Otto, a. (2019). Bootstrap. Retrieved from <https://getbootstrap.com/>
- [27] Gaston, K., & O'Neill, M. (2004). Automated species identification: why not?. *Philosophical Transactions Of The Royal Society Of London. Series B: Biological Sciences*, 359(1444), 655-667. doi: <https://doi.org/10.1098/rstb.2003.1442>
- [28] Fahlgren, N., Gehan, M., & Baxter, I. (2015). Lights, camera, action: high-throughput plant phenotyping is ready for a close-up. *Current Opinion In Plant Biology*, 24. doi: <https://doi.org/10.1016/j.pbi.2015.02.006>


9 Appendix:





Table 1: Literature review





Dataset Type key:

Leaf =  Plant =  Vegetation:  Flower:  Animal: 

Year	Title	Institution	Type of Data	Algorithms	Results/Accuracy
2018 	Automated plant species identification— Trends and future directions	Waldchen J, Rzanny M, Seeland M, Mader P	General summary of all types of data & algorithms	- Many	Deep Learning CNN's are most accurate
2018 	UAVs and Machine Learning Revolutionising Invasive Grass and Vegetation Surveys in Remote Arid Lands	Juan Sandino , Felipe Gonzalez, Kerrie Mengersen, Kevin J. Gaston	342,626 samples were extracted from the obtained data set and labelled into six classes for grass and vegetation	Algorithm 1 : Detection and segmentation of invasive grasses using high-resolution RGB images.	96.75% and 96.00% for single mapping of buffel grass and spinifex, respectively, and a multiclass detection rate of 96.54%.
2018 	Plant Classification Using Artificial Neural Networks	Luciano D. S. Pacifico, Valmir Macario, Joao F. L. Oliveira	Fisher's Iris Plant, The Wheat Seed Kernels data set, The 100 Plant Leaves data set	Multi-Layer Perceptron (MLP) artificial neural network trained with Backpropagation algorithm	leaf margin and leaf texture have been combined, the algorithm was able to achieve an average accuracy of 97.16%

2017 	Data Augmentation for Plant Classification	Pornntiwa Pawara, Emmanuel Okafor, Lambert Schomaker, and Marco Wiering	datasets for three plant classification problems: Folio, AgrilPlant, and the Swedish leaf dataset	AlexNet and GoogleNet trained from scratch or using pretrained weights	results show that the CNN methods with particular data augmented datasets yield the highest accuracies
2017 	Visual Classifier for Invasive Species	Elliott Jobson, Andres Hernandez, Stanford University	pre-trained on on the ImageNet dataset and subsequently trained on various augmented versions of the limited dataset provided	VGG-16 CNN architecture,	
2017 	Deep Learning for Plant Identification in Natural Environment	School of Information Science and Technology, Beijing Forestry University, Beijing 100083, China	BJFU100 dataset is collected from natural scene by mobile devices. It consists of 100 species of ornamental plants in Beijing Forestry University campus.	The Deep Residual Network: 26-layer ResNet,	ResNet26: 91.78% accuracy
2017 	Plant species classification using flower images—A comparative study of local feature representations	Marco Seeland, Michael Rzanny, Nedal Alaqraa, Jana Waldchen, Patrick Mader	3 different datasets: the Oxford Flower 17, the Oxford Flower 102, Jena Flower 30.	OpponentSIFT.	94% for OF17 dataset
2017 	Deep machine learning provides state-of-the-art performance in image-based plant phenotyping	Michael P. Pound Jonathan A. Atkinson Alexandra J. Townsend Michael H. Wilson Marcus Griffiths Aaron S. Jackson Adrian Bulat Georgios Tzimiropoulos Darren M. Wells Erik H. Murchie Tony P. Pridmore Andrew P. French	2 Datasets: Root system architectural traits were extracted from images of 2697 seedlings using the RootNav software (RootNav, RRID:SCR_015584) to produce images Shoot System: dataset contained 62 118 images, of which 49 694 were training images, and the	The shoot CNN contains more layers to accommodate the larger input image size. It also includes increased feature counts in deeper layers to address the more challenging classification task posed by the shoot images. Both networks end in neural network classification layers -The Caffe Library (for CNN iterative learning)	(>97% accuracy) for root and shoot feature identification and localization

			remaining 12 424 were used for validation.		
2017 	LeafNet: A computer vision system for automatic plant species identification	Pierre Barréa, Ben C. Stöverb , Kai F. Müllerb , & Volker Steinhagea	LeafSnap, Flavia and Foliage dataset	Deep Learning CNN	LeafSnap = 86%, Flavia =95.8%, and Foliage dataset = 97.9%
2016 	Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification	Srdjan Sladojevic, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic	A database containing 30880 images for training and 2589 images for validation. All the images collected for the dataset were downloaded from the Internet, searched by disease and plant name. Images were grouped into fifteen different classes.	CNN - CaffeNet architecture is considered a starting point, but modified and adjusted to support our 15 categories (classes).	Experimental results on the developed model achieved precision between 91% and 98%, for separate class tests, on average 96.3%.
2016 	Towards Automatic Wild Animal Monitoring: Identification of Animal Species in Camera-trap Images using Very Deep Convolutional Neural Networks	Alexander Gómez1, Augusto Salazar and Francisco Vargas	The Snapshot Serengeti contains images of 48 animal species(unbalanced). 26 classes were selected for classification	CNN Caffe	88.9% of accuracy in the Top-1 and 98.1% in the Top-5 in the evaluation set
2016 	Using Deep Learning for Image-Based Plant Disease Detection	Digital Epidemiology Lab, EPFL, Geneva, Switzerland & Department of Entomology, College of Agricultural Sciences, Penn State University, State College, PA, USA	PlantVillage dataset: 54,306 images of plant leaves, which have a spread of 38 class labels assigned to them. Each class label is a crop-disease pair. Used 3 variations of dataset: colour, greyscale, leaves segmented	AlexNet::TrainingFromScratch::GrayScale::80–20 GoogLeNet::TransferLearning::Color::80–20)	AlexNet=85.53% GoogLeNet: 99.34%

2015 	A Novel Method of Automatic Plant Species Identification Using Sparse Representation of Leaf Tooth Features	Taisong Jin , Xueliang Hou, Pifan Li, Feifei Zhou ¹	Constructed, as an image dataset, a total of 700 leaf images from eight plant species.	BP neural network, K-NN, proposed classifier	BP neural network 73.2±3.3 K-NN 72.3±2.5 Our proposed classifier 76.3±3.4
2013 	Automatic classification of legumes using leaf vein image features	Mónica G. Larese, Rafael Namías, Roque M. Craviotto, Miriam R. Arango, Carina Gallo, Pablo M. Granitto	Three legume species are studied: soybean, red and white beans. The leaf images are acquired using a standard scanner	SVM (Gaussian kernel), SVM (Linear kernel), Random Forests, PDA classifier	87% with the PDA classifier =scanned leaves, 9 % of accuracy for soybean vein characteristics
2013 	Leaf based plant identification system for Android using SURF features in combination with Bag of Words model and supervised learning	Quang-Khue Nguyen, Thi-Lan Le, Ngoc-Hai Pham	Flavia database	SURF features in combination with Bag of Words and supervised learning	95.94%
2005 	Mapping invasive plants using hyperspectral imagery and Breiman Cutler classifications (Random Forest)	Rick L. Lawrence, Shana D. Wood, Roger L. Sheley		Random forest	84% (Khat= 0.56) for the spotted knapweed site (Table 1) and 86% (Khat= 0.62) for the leafy spurge site