Table ­of Contents

[1.Introduction 3](#_Toc69650453)

[1.1 Introduction to problem 3](#_Toc69650454)

[1.2 Motivation and targeted audience 4](#_Toc69650455)

[1.3 Aims and objectives 5](#_Toc69650456)

[1.4 Deep neural network 7](#_Toc69650457)

[1.4.1. Introduction to deep neural network. 7](#_Toc69650458)

[1.4.2. Inspiration for neural networks. 8](#_Toc69650459)

[1.4.3. How does neural network learn? 9](#_Toc69650460)

[1.4.4. Different types of neural network. 11](#_Toc69650461)

[1.4.5. Machine learning algorithms for classification. 12](#_Toc69650462)

[2.Literature Review 13](#_Toc69650463)

[2.1 Introduction 13](#_Toc69650464)

[2.2 Hand crafted features 14](#_Toc69650465)

[2.3 Plant classification using artificial neural networks 14](#_Toc69650466)

[2.4 Plant lead recognition using shape feature and colour histogram with k- nearest neighbour classifier. 15](#_Toc69650467)

[3.Methodology 16](#_Toc69650468)

[3.1 Frameworks and software installation 16](#_Toc69650469)

[3.2 Data collection and pre-processing 16](#_Toc69650470)

[3.3 Data normalization and splitting 19](#_Toc69650471)

[3.2.1 Overfitting And Underfitting 20](#_Toc69650472)

[3.4 Designing the keras sequential model 21](#_Toc69650473)

[3.4.1 Layers of the model 21](#_Toc69650474)

[3.4.2 Compiling the model 23](#_Toc69650475)

[3.5 Sequential model with convolutional layers 25](#_Toc69650476)

[4.Evaluation & Testing 27](#_Toc69650477)

[4.1 Comparing Models 27](#_Toc69650478)

[4.2 Classification Report 27](#_Toc69650479)

[4.3 Confusion Matrix 27](#_Toc69650480)

[5 Discussion 27](#_Toc69650481)

[5.1 achievements of the research project 27](#_Toc69650482)

[5.2 defincensies of the research project 27](#_Toc69650483)

[6 Project Evaluation 27](#_Toc69650484)

[7 Conclusion 27](#_Toc69650485)

# Introduction

## 1.1 Introduction to problem

Plants underpin life on earth, they are fundamental to humankind as most of them carry information necessary for the progress of human society. Majority of the products we utilize today are from plants. Food and oxygen for the entire biosphere, biofuel, essential medicines and the significant role they play in climate regulation. We have till date documented Over 30,000 plant species which provide essential Uses Figure 1.1 . Agricultural yield and its products play a significant role in the economy of various countries.

Approximately 390,900 plants species have been discovered by botanists and 2000 new plant species are discovered each year ( Shreya Dasgupta, 2016). 40% of the plants species today are at the threat of extinction (Open Access Goverment, 2020) due to deforestation, change in climate and loss of habitat. Plant species are described as a “treasure chest” of medicines and food, and they have in the past demonstrated to assist humanity in tackling complex challenges. “The planet may be losing plant species more quickly before we can find, name and study them. This could have severe consequence’s as some food crops are resilient in the face of climate change” (Pavid, 2020). Discovering, defining, classifying and naming a species has so become a critical task, if we do not know a species exist we cannot protect it (Pavid, 2020).

Classifying and documenting them has become critical to progress in sectors of Botany, Pharmaceutical and Agriculture. The pace of species identification needs to be accelerated. If we do not move fast One third of the plant species could be gone in 50 years (University of Arizona, 2020). We are so, in a race against time to find, inspect and document Plant species before they go extinct.

Chart, bar chart

Description automatically generated

Figure 1.1 - Plant species documented and their uses.

## 1.2 Motivation and targeted audience

Plants vary greatly in diversity, the difficulty to classify plants is greatly enhanced as many species show resemblances in their structure Figure 1.2. Botanists are the people who study plants, by studying their morphological features (size, shape, margin and colour), they are able to classify the plant to a particular species. It would be a dreadful or even an impossible task to classify all plants, considering the volume of species that could be between 5 - 50 million (Stuart Pimm, 2010).

Encoding pre-defined morphological features of a leaf to a computer vision system by botanist has been applied in the past, but this method isn’t promising as this depends upon the ability of computer vision experts (Karim & Beikmohammadi, 2018, pp. 21-26) and Certain methods that have been used before only perform good on a particular set of datasets to a specific species (Zhang, et al., 2015, pp. 2143 - 2150). Several studies and research have been conducted in this area, but we still haven’t been able to produce a definitive solution and thus plant classification remains a challenging and unsolved question.

Deep learning has demonstrated great potential and produced promising results with its image processing, object detection and speech recognition techniques. It’s been deployed successfully in various areas to solve complex problems and has now entered the agriculture and botany sector. Advancements in deep neural network (DNN) has led to the development of Convolutional neural network (CNN) and transfer learning. CNN offers to extract high level features from images, thus saving time for experts to predefine morphological features or spend time on developing feature extractions methods. CNN does require a large dataset to produce good performance, using a small dataset could result in overfitting (mode heavily relying on training dataset) to overcome this problem, transfer learning is used at times. In transfer learning we use a pre-trained model (features learned during basic model training) (Karim & Beikmohammadi, 2018).

With the developments of deep neural network and the plant classification problem still un-solved it would be interesting to comprehend what results we obtain when we apply different models of deep neural network to solve this problem.

A plant classification system would accelerate the rate at which species are being identified, it would assist botanist In faster recognition and detection of endangered species. Some regions require our immediate attention which contain multiple species on the verge of extinction, we could so dedicate our resources to such areas and document them before it’s too late.

Such a system could be taken and developed into a web app or a mobile application which would help people who are inquisitive to know about the plants they grow or students who are aspiring to become botanist to know about its species, so the need of for a botanist would be reduced significantly. Agriculture can be made more cost-effective and environment friendly by spraying herbicides only on weeds (Kaya, et al., 2019, pp. 2-3). Controlling deforestation, remote sensing for farming and food safety are other area which would benefit from such a system.

A picture containing fabric

Description automatically generated

ssss


Figure 1.2 - Plants sharing similarities

## 1.3 Aims and objectives

**Primary aim.** The primary aim of this research will be to find the most accurate deep neural network model to classify plants by their species, which could then be used further to develop a plant classification system comprising of a website or mobile application.

**Objectives of research.**

* Developing different types of deep neural network (DNN) models.
* Training these models on a dataset containing features extracted from images of leaves.
* Using a testing dataset to evaluate the accuracy of the models.
* Analysing outputs and discussing Results.
* Providing a conclusion on which model to apply for plant classification.

The methods employed to evaluate the performance of our model is the following : -

* Using a validation dataset, A validation dataset will not be part of our training dataset and so is held back from the training process. It provides an unbiased estimate of the model’s performance after each epoch Figure 1.3. Passing the dataset forward and backward once through the neural network is called an Epoch.
* Using Learning curves Figure 1.4, Learning curves are used as diagnostic tool in machine learning they will demonstrate to us how the error change as the model trains or learns from the trained dataset. I would be using matplotlib pyplot library to draw training and validation learning curve, this well disclose how well the model is learning or help in diagnosing overfit and underfit models.
* A classification report 5which would assist us in measuring the quality of predictions the model makes when we use it on a testing dataset. For this I would be using sklearn metrics library.
* For summarizing the performance of our model, I would also be using a confusion matrix where applicable Figure 4. A confusion matrix provides a clear idea of what mistakes or errors is the model performing and what is getting correct, In other words a classification report summarizing correct and incorrect predictions. For this I would be using sklearn metrics library.

A picture containing table

Description automatically generated

Figure 1.3 - Validation accuracy after each epoch

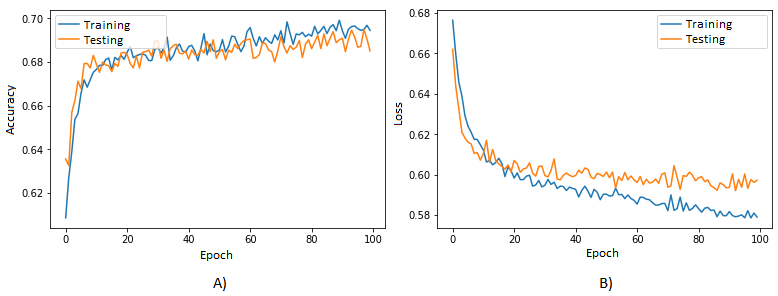


Figure 1.4 - Learning curves generated from model history

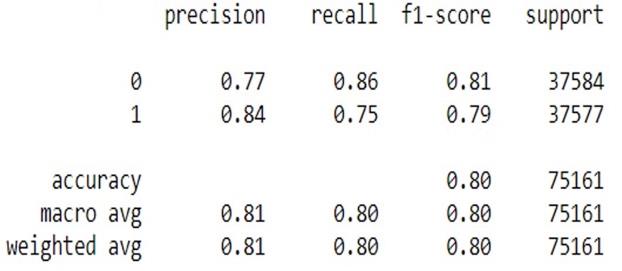


Figure 1.5 - Classification report for each class

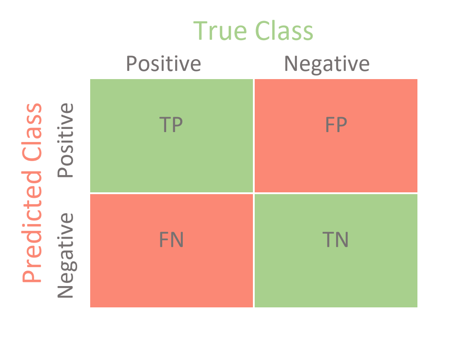


Figure 1.6 - Confusion matrix for evaluation

## 1.4 Deep neural network

### 1.4.1. Introduction to deep neural network.

The most crucial part of the human body is the brain. If the human body is described as a computer, the brain can be called the CPU of the human body which helps us in performing all the daily functions, consisting about 86 billion neurons. The brain is built of fundamental unit called neurons which is chiefly responsible for receiving sensory input and sending appropriate motor responses to the muscles (Woodruff, 2001). Each Neuron is connected to other neurons, which forms a complex network of interconnected neurons responsible for transmitting information.

Artificial neural network is a system which is designed to mimic the brain, the way we analyse and process information, it is based on the study of artificial intelligence to solve problems that would be difficult for human beings to solve in a short period of time example number crunching. An artificial neural network (ANN) system cannot mimic a human brain completely it can only be programmed to perform certain behaviours or functions the brain performs.

Machine Learning is the branch of artificial intelligence which focuses to program themselves from the input data it receives, it is able to improve its accuracy on its own by working with more input data over time. It provides Artificial intelligence the power to solve problems based on data, and so artificial neural networks are examples of machine learning algorithms (Ciaburro & Venkateswaran, 2017, pp. 1-5).

Deep Learning is a branch of machine learning which is primarily focused on algorithms centred around artificial neural networks. It consists of complex set of layers of neurons interconnected to each other. Each layer of these neurons provides us with more processing. We can apply deep learning to difficult tasks such as image identification, classification and handwriting detection.

Text, chat or text message

Description automatically generated

### 1.4.2. Inspiration for neural networks.

The manner in which our brain processes information or its overall functioning served as an inspiration for the foundation of neural networking. Our brains are constantly processing enormous amounts of data which it receives from different sensory organs predominantly the eyes. Information is carried by neurons and transmitted from one neuron to another applying a flip-flop logic. when a certain threshold mark is surpassed the neurons fire and the information is transmitted to the other neuron or they do not fire at all; it is either 1 or 0.

**Similarities**

Structure of a neuron Figure 7 : -

* Dendrites : Branching out from the cell body they are the fibrous root like structure. They form the entry or receiving point for the electrical signals that are received from adjacent neurons.
* Axon terminals : Outgoing part of the neuron, which transmits electrical impulses to the next neuron.
* Cell body : also called the soma, upon receiving electrical inputs from the dendrite it decides what action to take.

Diagram

Description automatically generated

Figure 1.7

Structure of an artificial neural network (ANN) Figure 8 : - A neuron is the basic structure of an ANN.

* Input Layer : it brings in the initial or input data into the system and passes it on to the other layer for processing.
* Hidden layers : this layer consists of the linear calculations and activation functions which decide whether to activate the neuron, fire or not.
* Output Layers : after the computation in the hidden layers, the final result is then passed on to the output layer so the user can view them and make a decision.

Chart, diagram, bubble chart

Description automatically generated

Figure 1.8

Computers have been surpassing humans in various domains, such as huge numerical calculations, classification and problem solving but there are some areas where brains have been outperforming machine’s. A toddler can recognize his/her mother in a huge crowd, but a computer wouldn’t be able to do this, hence serving as an inspiration to develop a loosely decentralized architecture which functions like a brain (Ciaburro & Venkateswaran, 2017).

### 1.4.3. How does neural network learn?

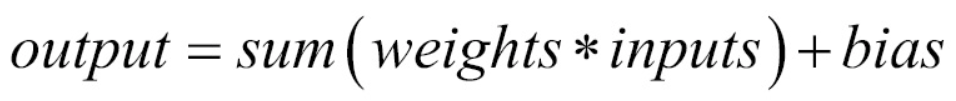
The Neurons in neural network function like a CPU, on receiving an input they perform mathematical operations on it generating an output which is then passed on to the next layer of neuron. By computing the output from all the neurons, the network is able to make a deterministic calculation, which is passed on to the final output layer for the end user to view.

Neurons are interconnected with each other by a weight (a number). Weights are the most essential part of a neuron, as the value they hold effects the output that the neuron will pass to the next one. If the input we receive are x1,x2 and x3 then the weights that will be applied to them will be represented as w1,w2 and w3  (Ciaburro & Venkateswaran, 2017). When we train our model for the first time these weights are set randomly.

A picture containing text

Description automatically generated

Where ‘I’ will be the number of inputs the neuron receives. A Bias value is also added which allow us to shift the activation function either to the left or to the right. The entire processing done by a neuron can be expressed as : -



The activation function is applied on this output which decides whether the neuron should be fired or not.

Diagram

Description automatically generated

This flow of information from input layer to hidden layers and finally to the output layer is called forward propagation.

Just like human beings learn from mistakes or by receiving feedback deep neural network learn the same way by a process called backpropagation. Once the output reaches the final layer, the error is calculated ( predicted output - original output ). The error is then used to re-calculate the weights and biases that was used in forward propagation. To calculate the new weight amount neural network, apply gradient descent.

Gradient descent is an optimization algorithm which is used to find the values of a function’s parameters, that minimizes the cost function as far as possible (Donges, 2019).Backpropagation is performed after each dataset iteration.

By moving forward and backward considerable number of times, readjusting the weights and biases, the neural network is able to reduce the difference between predicted output and actual intended output, and so the model will now have a good fit for the data and could provide us with better accurate results.

### 1.4.4. Different types of neural network.

There are several different types of neural networks Figure 9 each with their unique set of features or characteristics, providing them their own strengths. I would be explaining some of the important ones below : -

A picture containing scatter chart

Description automatically generated

Figure 1.9

**Perceptron** Originally coined by Frank Rosenblatt in his paper (ROSENBLATT, 1958). There are two types of perceptron: Single-layered and multi-layered. A single-layer neural network is also called feed forward neural network, consisting of an input layer and an output layer. The hidden layer is absent in a single perceptron model. The architecture of a multi-layer model is similar to a single layer and consists of one or several hidden layers. Multi-layer also performs back-propagation (adjusting weights and biases) unlike single-layer perceptron. In both of them the neuron takes an input, aggregates it or calculates the weighted input and then uses it on an activation function, if the output from the activation function exceeds a certain threshold the neuron will be activated or else remain in an inactive state; it will be either be a 1 or a 0. *Applications* : - Computer vision, speech recognition and classification.

**Recurrent neural networks** (RNN) First coined by (ELMAN, 1990, pp. 179-211). In a recurrent neural network, the output we get from a layer is saved and is then fed back to the input. The first layer remains the same as in a feed forward, but the subsequent layers are the place where RNN takes place. Each neuron remembers some information it had in the previous step, so each neuron processes information as well as stores in memory the previous value which is then used later. RNN are useful when decision or values from the past can influence the current ones (Tch, 2017). A slow computational speed remains the main issue of such a system, cannot re-collect information from several states back and cannot predict the future by the current state. *Applications* : - Speech recognition and synthesis, rhythm learning etc.

**Convolutional neural networks** (CNN) Lenet a handwriting digit recognizer was developed by YannLeCun and his associates, which was later given the nameConvolutional neural network. A CNN is primarily used to solve problems associated with images but can also be used on audio. They employ a variation of multilayer perceptron’s and would contain more than one layer of convolutional layers (Anukrati Mehta | Jan 25, 2019). Kernels or filters are the fundamental part of a CNN, they are used to extract features from an image such as colours, edges and corners and the resulting output is a distinct feature map from each kernel. These feature maps are then downsized to a smaller matrix by a process known as pooling. The final pooled matrixes or feature maps are then used as inputs, just like a fully connected neural network. *Applications* : - Identify faces, streets, tumours, Image recognition and Video Analysis.

### 1.4.5. Machine learning algorithms for classification.

why are deep neural networks are used for classification when there are several other algorithms which provide effective ways for classification.

Given an example input data to a pre-trained model, the model is able to predict which object the class belongs to, this is known as classification. Machine learning is being used for several classification problems such as image, speech and handwriting recognition. In some cases, a neural network might be an overkill as there are other classification algorithms which might be more suitable to solve a problem, for this we should be aware of them, their pros and cons.

**Logistic regression** A machine learning algorithm. It is Based on the concept of probability and is a predictive analysis algorithm. To predict the correct output, it uses logistic regression equation. Pros : - It is largely used in a situation when the data input could have just two outputs: 1 or 0 and when the input variables are well known. Cons : - When the relationship between input variables are not known or they are too complex, their effectiveness decreases.

**Naive Bayes Classifier** the classifier is based on Bayes theorem, by applying this theorem we would be able to find the probability of ‘A’ happening given that ‘B’ has occurred (Gandhi, 2018).

A picture containing text

Description automatically generated

Pros : - Performing multi-class predictions, fast and easy to predict on a test data set (ray, 2017). Cons : - Not Good at being an estimator, hence the probability outputs are not taken seriously at times.

**K-Nearest Neighbour (KNN)** Data Points are classified by analysing the nearest neighbours by storing instances from the training set. It makes the use of lazy learning (does not train on complete dataset) (MissingLinkAI, 2019). Pros : - Easy for implementing the KNN model and simple to understand. Provides highly effective and competitive results. Cons : - Unsuitable for high dimensionality problems.

*Deep neural network unique ability to create functions dynamically which are able to analyse complex data, predict outputs and mimic the human brain separates it from the rest of the classifier’s.*

Introduction to the rest of the report , report structure section , summary of each chapter 2-3 sentences

# 2.Literature Review

## 2.1 Introduction

Machine learning methods to automatically classify pants into their species has shown promising results (K.B.Shobana & P.Perumal, 2020), to solve the problem of plants going under extinction before they can be documented. Improvements and further research in AI and machine learning have led to the development of deep neural networks which has demonstrated high potential and promising outcomes for processing images and analysing data. Showing success in various other fields deep neural network has now entered the field of agriculture (Kamilaris & X.Prenafeta-Boldú, 2018), therefore It’s different architecture and models are now gaining more attention.

Previously research has been conducted to solve the plant classification problem by applying different solutions, some of them have gained success while some of them can be improved further with deep neural network (DNN). Section 2.2, 2.3, 2.4 and 2.5 contain research which show how this research relates to previous ones, gaps in their studies, approaches which are no longer used or outdated, difference in the views of author and how we can apply DNN to fill overcome these concerns.

## 2.2 Hand crafted features

Studies have been performed by selecting features from images of leaves such as texture, shape, margins and venation. The authors (Neto, et al., 2006) applied elliptic Fourier and leaf shape analysis for plant species classification using their shapes. This research (Chaki & Parekh, 2011) proposed an automated system for plant classification by using two different modelling methods: moments - invariant and Centroid – Radii. Histogram of curvature over scale has been proposed by (Hall, et al., 2015) and (Kumar, et al., 2012). Applying hand crafted shape has also been proposed by (Hall, et al., 2015).

Texture, shape and venation most of the hand-crafted studies are based on these features. The solutions using these features have shown promising outcomes, but they heavily rely on the set features selected from a dataset, therefore when dealing with another dataset, it is not necessary we will find these features in them. To identify and analyse hand crafted features there is a requirement of botanist, which may not be available easy and so slows down the process even more. Due to these disadvantages, Researchers are now trying to adopt a deep learning approach which eliminates the presence of an expert and will be able to identify and extract features regardless of the difference in datasets.

## 2.3 Plant classification using artificial neural networks

The authors of this research (Pacifico, et al., 2018) have used an architecture of an artificial neural network known as multi-layer perceptron with backpropagation (MLP-BP). Their views from the studies mentioned in 2.1 differ, mentioning that such studies (section 2.1) are more task dependent and are heavily influenced by the set features extracted from them which leads to a data set bias problem, also some of this research (section 2.1) only focuses on the proposed classifier by them but they do not compare their models with previous classifiers or models from their literature review. The authors of this research do agree with another research (Kumar, et al., 2012) that manually identifying plants are more susceptible to error’s and to solve the problem of plant extinction an automatic system is the most promising solution.

This research has been able to demonstrate that by applying neural networks they were able to classify plants in an unbiased fashion and achieve an accuracy of 83.29%, and when compared with other machine learning classifiers their model of MLP-BP showed much better accuracy. By applying image pre-processing, data augmentation and by using a much bigger dataset we could increase the accuracy and get better results. It would be interesting to learn what changes we get when we apply these methods in our deep neural network model.

## 2.4 Plant lead recognition using shape feature and colour histogram with k- nearest neighbour classifier.

In the research (Munisami, et al., 2015) the authors applied a machine learning classifier k-nearest neighbour classifier (section 1.4.5) on a dataset consisting of 20 different images of each species, a total of 32 plant species. They were able to achieve an accuracy of 83.5% and by further enhancing the images by colour histogram the accuracy was increased up to 87.3%.

The authors have spent considerable amount of time in designing the image processing techniques and which features need to be extracted from the leaf, so paying less attention to the classifier itself. They have used shape as the key feature to extract and state that in future they would be using shape and vein feature combined to classify plants, as mentioned (section 2.1) this makes the classifier too much dependent on the selected feature set.

With the further enhancement of deep neural network, convolutional neural networks have been introduced which allows researchers to focus primarily on the classifier and leave the feature extraction techniques to the network itself saving the need of an expert to be present or to design several layers of features extraction techniques before passing the data to a classifier.

# 3.Methodology

## 3.1 Frameworks and software installation

In this research several frameworks or software have been used. These will assist in writing, executing code, data collection, pre-processing, normalization etc.

Jupyter notebooks - An open-source web application that will allow us to create, edit and share documents. it is mainly used for data visualization and machine learning. The key advantage of using jupyter notebooks is that it can be executed on a local desktop.

Google Collaboratory - Allows anybody to write and execute python code on the browser. Will save time as no pre-configuration would be required, streamlines the process of sharing and as neural network or machine learning training requires GPUs, google Collaboratory provides free access to GPUs.

Pandas - Mostly used by data scientist. Allowing for data exploration and analysis, will be used to read csv file and convert to a data frame or table. This research will be using pandas to read the data from a csv file, manipulating it and converting it to multidimensional array.

NumPy - A library or package used for scientific computing in python, allows to convert data frames to multidimensional arrays. In this research it will be holding all the training data and its associated labels or classes. It makes the process of performing normalization on the data set much simpler.

Keras - Library will be used to build the deep neural network and its layers such as Dense, maxpool1d, dropout, input and output layers. These layers and their functioning would be described in more detail in the model implementation section.

Matplotlib - A plotting library for python, providing the ability to visualize data, evaluating and analysing the output results from our model.

Apart from the libraries and frameworks mentioned above, there are other libraries which are being used in this research, they will be described as we apply them to build the model.

## 3.2 Data collection and pre-processing

The dataset this research will be using is obtained from (Mallah, et al., 2013). It consists of 1584 images of leaves. A total of 99 species have been identified with each species containing 16 images. A colour image was first captured, Followed by performing image segmentation process on each one of them, it consisted of applying global thresholding on grey scale images and then manual correction for any spurious segmentation (Mallah, et al., 2013). These images Figure 10 are then used for the feature extraction process.

Shape, arrow, circle

Description automatically generated

Figure 0.1 – Grey Scaled Images (a) Acer Campestre (b) Acer Cincinatum  
(c) Acer Mono (d) Acer Palmatum (e) Alnus Rubra (f) Alnus Sieboldiana  
(g) Betula Austrosinesis (h) Eucalyptus Urnigeria (i) Ilex Aquifolium  
(j) Ilex Cornuta.

Three types of leaf features are targeted - shape of the leaf, interior texture feature and margin features. To extract each feature a different methodology has been applied. For the shape of the leaf, a shape descriptor function is being used called Centroid Contour Distance Curve (Wang, et al., 2000) which represents a two-dimensional area or boundary. Textures for the plant was obtained using Gabor Co-occurrence’s (Cope, et al., 2010), Gabor filter is applied to 1024 randomly selected windows on the leaf surface and each of the window is quantised to a 64-size vector. For the margins of a leaf a median filter is applied to the image, which presents a smoothed version of the leaf followed by selecting points from the image which encompasses the full outline. For each feature (shape , margin and texture) a 64-size vector is provided for every image of leaf.

Importing the dataset into google Collaboratory, pandas library is then used to convert the csv file into a data frame structure which consists of 990 rows and 194 columns Figure 11.

Table

Description automatically generated

Figure 0.2

The data frame contains an id column which maps to the image of the leaf, the species column identifies the species the leaf belongs to, after which there are 64 columns for margin, texture and shape points on each leaf. Id column will not be used; therefore, they will be dropped from the data frame.

**Categorical Encoding** Any dataset contains multiple columns which are formed of numerical and categorical values. As a computer’s only understand numbers not text, any data field containing categorical value will have to be converted, in this dataset the species column has been converted to a numerical value which the neural network can process and understand. The two most widely used techniques for categorical encoding are label encoding and one hot encoding, this research applies label encoding where each species is given a numerical value based on alphabetical ordering Figure 12.

Table

Description automatically generated

Figure 0.3 - label encoding

After assigning it a numerical value, the species column is removed and saved in a NumPy array, the training dataset now only contains values which are used for the final training Figure 13.

Table

Description automatically generated

Figure 0.4

## 3.3 Data normalization and splitting

Dataset normalization or standardization is a process of making sure that all the data points gathered are on the same scale ­­­­­and that every data type has the same content and format. In a typical normalization process the numerical data fields are scaled down to a point between 0 and 1. The most used method to normalize data fields are given is given by the formula by subtracting the mean from each data point or value and then dividing the difference by the standard deviation.

A picture containing text

Description automatically generated

This research uses scikit-learn Standard scalar method (Cournapeau, et al., 2007) to create standard scalar object. While creating the object, the dataset is passed to it which computes the mean and standard, and returns a scaled object followed by using the transform method which performs standardization and returns a transformed array Figure 14.

Table

Description automatically generated

Figure 0.5 - Dataset Normalized

Once the dataset has been normalized, it has to be split into three parts training, testing and validation. Each of these dataset serves a purpose during the different stages of the model development. *Validation dataset* will be used to provide an unbiased evaluation of the model performance while tuning its hyperparameter’s, the model views this data during the training process but never learns from it. *Testing dataset* contains data which will be used to provide an unbiased evaluation of a final model fit on training dataset (Brownlee, 2017). *Training dataset* contains all the sample data which the model uses to learn or train itself.

Data splitting ratio mainly depends on two factors, type of model being trained, and the total number of samples accumulated. A model which has few hyperparameters is simple to validate and tune and so the size of validation dataset can be reduced but if our model has many hyperparameters the size of the validation set should be increased (Shah, 2017). In many cases the size of train, testing and validation dataset is selected by training the model and analysing the output, which is different to each use case.

Table

Description automatically generated

In this research training, validation and testing ratio is 70:20:10. By using scikit-learn stratified shuttle split library (Pedregosa, et al., 2007), firstly a stratified shuttle split object is created by passing the testing size (10%) as a parameter which returns a stratified randomized object. This object is then used to call the split method on the training set and the species array created earlier both of them are splitted by 10% into four variables training set (X\_train , y\_train) and testing set (X\_test, y\_test).

### 3.2.1 Overfitting And Underfitting

The main purpose of having three types of dataset is to confirm if our model is able to generalize on data it has not seen before and predict accurate results. If the model isn’t able to generalize the data, the model is said to be in a situation of overfitting or underfitting

**Overfitting**

Occurs when the model learns the data present in the training dataset too well and start negatively impacting performance of the model on the new data or start performing poorly on the dataset it hasn’t seen before. During the training phase the model should be learning the relationships between the features presented and their associated labels (Brownlee, 2016).

Spotting if the model is overfitting is done by observing the metrics which are shown during the training phase. A validation dataset is split from the training set before and passed as an argument to the fit function. if the accuracy obtained is very good but the validation accuracy is poor, it is said the model is overfitting.

Other ways to reduce overfitting : - Adding more data to the dataset will add more diversity to the dataset and the model will be able to learn from a bigger diverse dataset so learning more features. Data augmentation is a technique where additional data is created reasonably by modified (cutting , cropping, zooming and flipping) helps to reduce overfitting. Reducing complexity of the model can also be done by reducing some neurons or layers from the model or using dropout layers (Deep Lizard, 2017) which have been used In both the models describes below.

**Underfitting**

It is the opposite of overfitting, occurs when the model is not able to classify data it was trained on. It can neither model the training data nor generalize the new data (Brownlee, 2017).

By observing the metrics during training phase, a low training accuracy and a high training loss it is said the model is underfitting.

Some of the steps that can be taken to reduce underfitting are – Increasing the complexity of our model which is opposite to overfitting, which can be done by increasing the number of layers and neuron used, using different types of layers in appropriate places. Reducing dropout layer in the model is also helpful to reduce underfitting (Deep Lizard, 2017).

## 3.4 Designing the keras sequential model

In this research the keras framework has been used, to develop the deep neural network model. It is an open-source library, consisting of components written in python used for neural networking. The model developed in this research uses keras sequential API that allows to create layer by layer (linear stack of layers) that will be used in the model. keras offers flexibility in creating complex models and all the different layers that will be required by the model are imported from the keras library as well.

### 3.4.1 Layers of the model

**Dense layer** The first layer used in the model, is a Dense layer. A dense layer is a neural networking layer connecting deeply, which means that every neuron present in its layer receives an input signal from the previous layer (Sharma, 2020). The first dense layer defined in the model is actually the second layer, as the input layer is created automatically by keras when the input parameter is passed to the first dense layer. The parameters used in the model are : -

* Units - The number of neurons to be used in a layer is specified here. The number of neurons to apply for a layer is arbitrary. 100 neurons have been applied for this layer of the model.
* Input Shape - As the name suggests, this parameter takes in the shape of the input data we pass to the model. This layer could be defined separately and then passed to the model or the dimensionality of the input shape could be passed as an argument, in the form of a tuple. (192,1) is the argument passed for the model. As the input data is 1 dimensional and the pre – processed data set contains 192 columns.
* Activation function – ReLU activation function has been chosen for all the layers in this model. it is the most common activation used in DNN as it is simple to implement and effective to overcome limitation (vanishing gradients) which can be experienced by Sigmoid or Tanh activation function’s (Brownlee, 2021).

**MaxPool Layer** A MaxPool1D layer has been inserted as the second layer. Max pooling is a sample-based discretization process, which downsizes an input representation in its dimensions, so allowing assumptions to be made about its features. It is mostly used in Convolutional neural networking over images as it reduces the computational load when going over larger images data. in this research it has been used, as it also reduces the problems of overfitting (Lizard, 2018). The ‘1D’ signifies the shape of the input data, which is in 1 dimension. Keras also offers 2d and 3d max pooling layers.

**Flatten layer** A flatten layer has been added to this layer of the sequential model. A flatten layer is used to flatten the input we receive from the previous layer. It assists us in coupling information that is present vertically as well as horizontally in the dataset being used.

**Dropout Layer** Deep neural network at times tend to overfit on a training dataset quickly with only a few examples. Groups of models with different configurations can decrease overfitting, but this would require additional models and will be costing computational power (Brownlee, 2018). We can use a single model and simulate having a large number of different architecture’s which can then be randomly dropped out during the training process. This process is called dropout which helps in reducing overfitting and improving generalization error. This research model applies a dropout layer with a rate of 0.25, the rate is the fraction of the inputs units or neurons to drop after each step during the training time.

The Last Layer or the output layer is a keras dense layer being used in the model. the number of neurons specified are 99, which are the different classes we expect as the outcome from the model or simply the 99 species our dataset contains.

Table

Description automatically generated

Figure 0.6 – Layers in DNN

Diagram

Description automatically generated

Figure 0.7 - Architecture of Sequential model.

### 3.4.2 Compiling the model

Once the layers with their input parameters have been defined, the model would then be compiled using the compile method. Three arguments are passed to the compile method : -

*Loss function* : In the context of an optimisation algorithm, the loss function is used to calculate the perfect weights for a neural network which is also known as the objective function. In neural network we are seeking to minimize the error, the value that is calculated by the loss function is also known as loss (Brownlee, 2019).

The choice of loss function depends on the activation function being used in the output layer; they both are connected. As for this research there are multiple classes to be predicted, the model would be predicting the likelihood of an example belonging to each class. For this we have chosen SoftMax as the activation function in the output layer and Cross entropy also called Logarithmic loss as the loss function.

*Optimiser* : A function which is used to change the weights and learning rate of the neural network being used, to reduce the loss. There is different optimiser offered by the keras framework; SGD, RMSprop, Adam, Adamax etc. For this model Adam (Adaptive moment estimation) has been used. It has a low memory requirement and work’s even with minimal tunning of the hyperparameters.

*Metrics* : A function which is being used to judge performance of the model. they are alike the loss function mentioned above, except the results are not used when training a model (Kears, 2019). The metrics argument holds the list of metrics that needs to be evaluated by the model during its training and testing phases. There are different kinds of metrics offered by keras Accuracy, Probabilistic, Regression and image segmentation metrics. The accuracy metrics class has been used for this model, which calculates how often predictions equal labels.

Once the model has been compiled, the training phase of the model begins. The model has been trained for 120 epochs, with a validation split of 20%. At each epoch data for the loss, accuracy, validation loss and validation accuracy are given.

During the beginning of our training phase, the loss value given is 445%, accuracy is less than 0.05% and a validation accuracy at 0.17%. In the last epoch, the loss value given is 0.014%, accuracy given is 92.23% and validation accuracy at 95%. In Figure 17 – Sequential Model Accuracy the accuracy for the entire training process has been shown, where the validation accuracy remains lower than train accuracy. In Figure 18 – Sequential Model Loss the loss function value has been shown for training process.

A picture containing shape

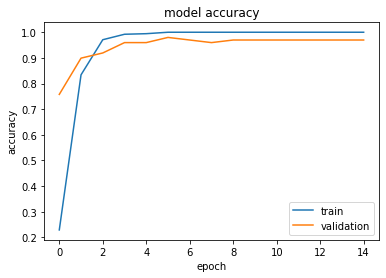
Description automatically generated

Figure 0.8 – Sequential Model Loss

Figure 0.9 – Sequential Model Accuracy

## 3.5 Sequential model with convolutional layers

Convolutional neural network was primarily developed for classification of images. the network learns an internal representation from a two-dimensional input, this process of learning is called feature learning. This way of learning can also be applied on one dimensional type of data.

The advantage of using a 1D CNN model is that they can learn from raw time series data directly and would not require manually hard coding input features, it’s compact architecture configuration makes it suitable for real time fault detection, it can work without any pre-determined hand-crafted features and efficiently trains on dataset with a limited size of dataset.

In the second model designed for classification, a 1D convolution layer has been applied as the second layer with 512 filters, and a kernel size of 1, the input shape is same as used above in the sequential model. Before applying the final output layer, 3 dense layers have been used to increase the accuracy of the model by increasing the connection between the layers and the weights between them. In Figure 19 - Layers of Sequential CNN all the layers applied with the number of neurons used have been shown and in Figure 20 - Architecture of Sequential CNN the flow of data from one layer to the other have been shown.

Table

Description automatically generated

Figure 0.10 - Layers of Sequential CNN

Diagram

Description automatically generated

Figure 0.11 - Architecture of Sequential CNN

For compilation of the model, an SGD optimiser has been applied also known as Stochastic Gradient Descent. It provides calculations much faster than gradient descent and batch descent. Data scientist have also claimed that performing one pass of SGD on a dataset can get better results on the expected loss and performs updates more frequently than any other optimiser (BHATIA, 2018).

A picture containing shape

Description automatically generatedA picture containing shape

Description automatically generatedAfter the model has been compiled, the fit function is used to train the model for 15 epochs with batch size of 16. In the first epoch the accuracy of model is 13%, loss at 425%, validation loss 142% and validation accuracy of 70%. The final values observed are accuracy at 96%, loss at 0.12% and a validation accuracy of 97%. In Figure 21 - Sequential CNN Accuracy the validation accuracy for training and validation dataset is shown and in Figure 22 - Sequential CNN loss, the loss for training and validation set is shown. The loss has decreased significantly from 1 – 4 epochs and accuracy has increased swiftly; the validation accuracy though remains lesser than the actual model accuracy offered.

Figure 0.12 - Sequential CNN Accuracy

Figure 0.13 - Sequential CNN Loss

# 4.Evaluation & Testing

In this section, both the models have been evaluated by Analysing their accuracy, loss, and validation outputs. The metrics that have been used evaluating are, model accuracy, model precision, recall and f1 score. These metrics assist in informing how well the model is behaving.

Using the testing dataset that has been split in section 3.3, it will be used on both the models that have been designed to classify plants. Keeping the testing dataset completely separate helps in giving an unbiased evaluation for the final trained model.

Reshaping the testing data to the input shape of the model, the predict method is then called with the testing data passed as the argument. The predict method return array of prediction’s (y\_preds)for each row of information inserted.

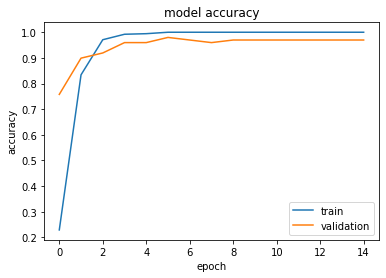
In section 4.1, the learning curves have been plotted, the values obtained from both the models have been analysed and their findings have been discussed.

In section 4.2, different classification metrics have been used on the testing dataset and the entire dataset as well to test the efficiency and performance the model offers.

## 4.1 Analysing Learning curves and output values.

Using matplotlib, the learning curves for both the models have been plotted using the history property which is returned after the training phase completes.

A picture containing shape

Description automatically generated

Sequential with CNN

Sequential model



Figure 0.1 - Metrics given for both models

**Discussion**

The accuracy obtained for sequential model with CNN layers and SGD optimiser is 3% greater than a sequential model, sequential model is still experiencing loss (3.53%) which is greater than CNN layers (0.11%), the validation loss offered by sequential model is high (30.41%) compared to CNN which is just at 8.95% . this low validation loss can be explained as the sequential model uses a convolutional layer and 2 additional dense layers. Validation accuracy is 2% greater for CNN model than normal sequential model.

Observing the validation accuracy and accuracy for both the models, it can be said that none of the models are experiencing underfitting or overfitting problems. Sequential model CNN offers better performance comparatively to sequential model.

## 4.2 Classification Metrics

To understand the different classification metrics being used, a confusion metrics need to be understood. A confusion matrix is a table which is used to evaluate the performance of a classification model, when it is used on testing data for which all the true values are known. There are 4 parameters which need to be understood : -

* True Positive (top-left ) : How many times did the model correctly classify a positive sample as Positive.
* False Negative (bottom-left) : When a model has classified a positive sample as negative or incorrect classification.
* False Positive (top-right) : When a model has classified a negative sample as positive.
* True Negative (Bottom-right) : Correctly identifying a negative sample as a negative.

Chart

Description automatically generated

Figure 0.2 - Confusion Matrix

**Accuracy metric** It defines how the model has performed across all the classes. it is of importance when all the classes being used for classification is of equal importance. Accuracy is calculated by the ratio between the number of correct prediction’s to the total number of predictions (Gad, 2021). It performs well there are equal number of samples belonging to each class that needs to be classified.

A picture containing text

Description automatically generated

Figure 0.3 - Formula to calculate accuracy metric.

Using the sklearn.metrics the accuracy method is called, passing two arguments : y\_test which contains all the correct predictions and y\_preds which contain all the predictions made by the model.

The accuracy returned is 93.23% on the sequential model and 97.63% on the sequential model with CNN layers.

**Precision metric** Also called the positive predictive value, it is the fraction of relevant instances among the retrieved instances. The precision accuracy reflects how accurate a model is in classifying samples as positive. Main aim of precision metric is to classify all the positive samples in a dataset as a positive and not misclassify a negative sample as a positive one.

A high precision relates to a low false positive rate. To calculate precision, we use the formula Precision = TP/TP+FP where TP is the number of true positive and FP is the number of false positive.

Text

Description automatically generated

Figure 0.4 - Formula to calculate precision metric

Using the sklearn.metrics the Precision method is called, passing the true values labels and predicted outputs from the model.

A precision score of 91% is obtained with a sequential model and 98% with a sequential model with CNN layers.

**Recall metric** Recall, which is also called sensitivity is the fraction of relevant instances that are retrieved. It cares only about the positive samples. Obtaining a high recall value shows that the model has predicted more positive samples. It is independent of how the negative samples are classified (Gad, 2021).

It is calculated as the ratio of the number of positive samples predicted to the total number of positive samples actually present in the dataset.

**Text

Description automatically generated with low confidence**

Figure 0.5 - Formula to calculate recall metric

The recall score obtained for sequential model is 91% and sequential model with CNN layer is 97%.

**F1 Score metric** The F1 Score metric measure the test accuracy. It is calculated from the precision and recall of the test; thus, it takes both the false positive and false negative into account. Mostly used when the developer wants to seek a balance between the precision and recall values. Range of f1 score is between 0 – 1. F1 score metric could be more useful when the testing data set being used has an un-even distribution in its classes.

Text

Description automatically generated

Figure 0.6 - Formula to calculate f1 score

The f1 score metric obtained for the sequential model is 92% and for the sequential model with CNN layers is 98.89%.

Table

Description automatically generated

Figure 0.7 - Metrics for both models

By analysing the outputs received Figure 29 from the metrics, sequential model with CNN layers offers a better efficiency or performance when tested with a testing dataset.

## 4.3 Classification report

Using scikit-learn classification report method, a report has been generated below which shows the Precision, Recall and f1 score obtained for each class present in the dataset, which is the 99 type of different type of species. after splitting the dataset into three halves test, train and validate, the number of samples in the test dataset is greatly reduced, there is only row of data available for each species. To test on much bigger dataset test, train and validate sets are combined together and then used to generate the classification report.

Table

Description automatically generatedTable

Description automatically generated

Table

Description automatically generated

The above classification report is generated when tested on the sequential model.

Table

Description automatically generatedTable

Description automatically generatedTable

Description automatically generated

The above classification report is generated when the dataset is tested on a sequential model with CNN layers.

**Discussion**

# 5 Discussion

## 5.1 achievements of the research project

## 5.2 defincensies of the research project

# 6 Project Evaluation

# 7 Conclusion