A specific model using leaf detection in machine learning performs better than a deep learning model.

Sivakumaran Sarvanan*(BSc (Hons) in Computer Science).

Abstract: - *Many computer vision tasks rely on (deep) neural networks, and aim to predict* on the image. *However, not all tasks require a deep learning model. With a machine learning approach, we can aim to determine natural groups or clusters of images without being constrained to a fixed number of (learned) categories. how to pre-process images, extract features (PCA, HOG), and group images with high similarity taking into account the goodness of the clustering later classification K-NN regression machine learning algorithm to find the similarity image. I will demonstrate the clustering of the leave data set. In deep learning approaches certain data apply in different models such as VGG16 and VGG19, Inception V3 and resnet152V2. Nowadays there are various pre-learned models for the classification tasks. The common theme is that all supervised models require a learning step where ground truth labels are used to learn the objects for the model. Some models can readily recognize hundreds of objects and this number is steadily increasing but the majority of domain-specific objects will likely remain unknown in pre-learned models. In such cases, the transition towards unsupervised clustering approaches seems inevitable. Building high-quality labeled datasets is a laborious task and favors the use of clustering approaches.*

Key words: -
K-Nearest Neighbours
Supervised model
Unsupervised model
Deep learning
Machine learning

*Author for Correspondence
Sivakumaran Sarvanan.
E-mail: sivakumaransarvanan@gmail.com

Temporary Instructor, Department of Computer Science,
University of Jaffna, Sri Lanka

Received Date: February 16, 2023 Accepted
Date: February 20, 2023

Citation: Sivakumaran Sarvanan. A specific model has better performance than a deep learning model using leaf detection in machine learning.

Introduction

In the pre-learned model, this number of trained data is steadily increasing but the majority of domain-specific objects will likely remain unknown in pre-learned models. Thus we aim to establish a software solution for plant leaf disease detection using image processing which can both increase the accuracy of work and reduce the need for a human workforce. Our study concepts, all the tasks in supervised modeling approaches such as using training models, why? and unsupervised models, how it?

In SriLanka, which had the most economic crisis then food items are most in demand. So agriculture is one of the most important sectors. The farmers constitute about 40% of the total population. They are interested in different crop plantations. Northern Province is almost Jaffna, especially for brinjal vegetable plantations. Brinjal has various varieties of our country. In our areas, especially “Thinnavelli purple”[3], it is a country. It satisfies our climate condition and produces a certain income through export Agriculture. It covers the second largest extent after the curry banana hardy plant compared to other vegetables grown in Sri Lanka.

Various efforts have been developed to prevent brinjal crop loss due to diseases. Disease management is a major risk in brinjal village. When if it is major diseases on brinjal leaves Bacterial wilt (pseudomonas solanacearum), Cercospora leaf spot (cercospora solani me-langenae), Tobacco mosanic virus (TMV), Collar Rot (Sclerotium molfsii). The conventional approach to disease detection and plant maintenance is human intervention in the majority of agriculture around the world. The detection and identification and the naked eye is a quiet, challenging task for farmers.

We will collect the data from brinjal village. These data can be classifying every disease and taking photos through mobile phone cameras. Then using google colab upload the data apply some pre-processing after can be using PCA-HOG methods segmented those data using those result can be extract the features to that images using those methods cluster healthy and unhealthy leaf lastly trained that data testing to test data classify using convolutional neural network. These various steps can be trained and find each and every step loss and accuracy using three different keras functions such as resnet152v2, Transfer Learning Inception V3 using Keras, Transfer Learning VGG 16 and VGG 19 using Keras and evaluate each of the functions. The use of disease detection techniques is beneficial to detect a plant disease at a very initial stage. We aim to use image processing

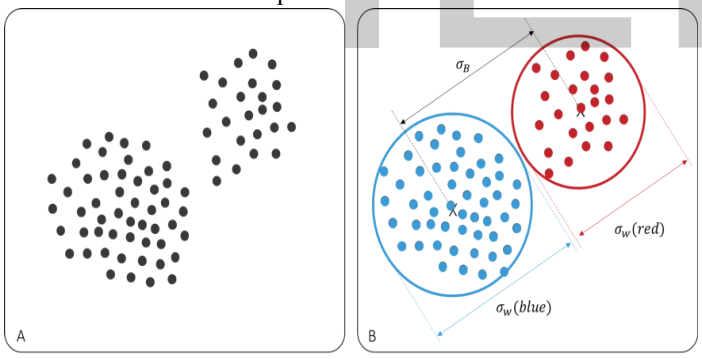
which can both increase the accuracy and timely of work and reduce the human work force.

In unsupervised learning PCA and HOG feature extraction, dbindex, silhouette score to clustering evaluation used cluster, hierarchical dendrogram clustering evaluation used cluster, t-SNE clustering evaluation used cluster and eliminate noise to clustering used DB algorithm and find similar image in training data same worked on testing data using based on the K-nearest neighbour
detecting based on significance after probability density functions 0.05 confidence level.

Details of previous work

The raw images needed to be processed as natural images tend to have a significant amount of background noise which can prove to be a factor later on in giving inaccurate results. To solve this issue, take these leaves in front on white sheet before taking photos and they utilize several filtering techniques in order to identify and extract the infected parts relevant to their research.

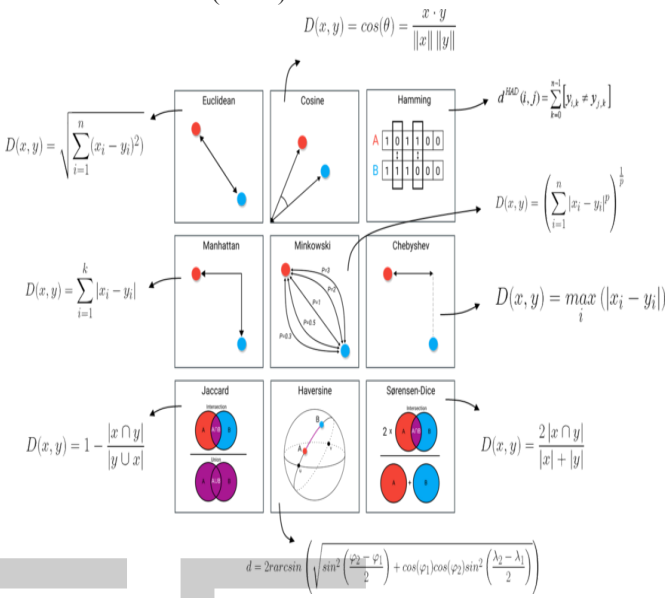
With unsupervised clustering [24], we aim to determine '*natural*' or '*data-driven* groups in the data without using prior knowledge about labels or categories. The challenge of using different unsupervised clustering methods is that it will result in different partitioning of the samples and thus different groupings since each method implicitly imposes a structure on the data. Thus the question arises; *What is a 'good' clustering?*.A depicts a bunch of samples in a 2-dimensional space. Intuitively we may describe it as a group of samples (aka the images) that are cluttered together. I would state that there are two clusters without using any label information. *Why?* Because of the distances between the dots, and the relatively larger 'gap' between the cluttered samples.



Clustering example

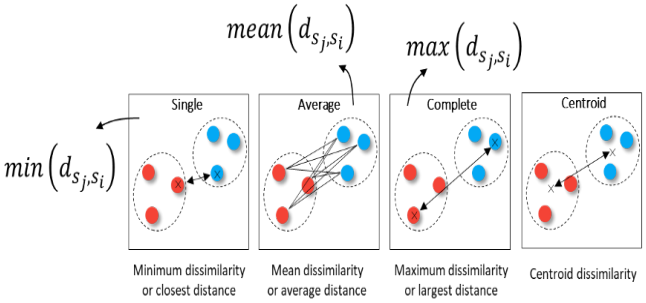
With this in mind, we can convert our intuition of '*clusters*' into a mathematical statement such as; the variance of samples within a so-called-cluster should be small (*within variance* σW*, red and blue*), and at the same time, the variance between the clusters should be large (*between variance,* σB) as depicted B. The distance between samples (or the intrinsic relationships) can be measured with *a distance metric* (e.g., *Euclidean distance*), and stored in a so-called *dissimilarity matrix*. The distance between groups of samples can then be computed using the *linkage type* (for hierarchical clustering).

The most well-known distance metric is the Euclidean distance. Although it is set as the default metric in many methods, it is not always the best choice. Understand the mathematical properties of metrics so that it fits the statistical properties of the data and aligns with the research. Properties of the data and aligns with the research. A schematic overview of various distance metrics is depicted in figure 3[1]. In the case of image similarity, the Euclidean distance is recommended when features are extracted using Principal Component Analysis (PCA) [2] or Histograms of Oriented Gradients (HOG).



Schematic overview of the most popular distance metrics

The process of hierarchical clustering involves an approach of grouping samples into a large cluster. In this process, the distance between two sub-clusters need to be computed for which the different types of linkages describe how the clusters are connected.



Linkage types

Briefly, Single linkage between two clusters is the proximity between their two closest samples. It produces a long chain and is therefore ideal to cluster spherical data but also for outlier detection. Complete linkage between two clusters is the proximity between their two most distant samples. Intuitively, the two most distant samples cannot be much more dissimilar than other quite dissimilar pairs. It forces clusters to be spherical and have often 'compact' contours by their borders, but they are not necessarily compact inside. Average linkage between two clusters is the arithmetic mean of all the proximities between the objects of one, on one side, and the objects of the other, on the other side. Centroid linkage is the proximity between the

geometric centroids of the clusters. Choose the metric and linkage type carefully because it directly affects the final clustering results. With this in mind, we can start pre-processing the images.

The technique used in the paper from Sladojevic, Arsenovic, Anderla, Culibrk and Stefanovic (2016) is a method for the identification of plant diseases using a trained deep convolutional neural network which is fine-tuned to plant leaves database [12]. Neural network is an information-processing model in machine learning and cognitive science, inspired by biological nervous systems. An artificial neuron is a processing unit with many inputs and one output. While artificial neurons may have many outputs, it will be considered only those with exactly one output. The CNN model used in this research works in a simpler way which aims to distinguish between diseased leaves from healthy ones. In any machine learning based project collection of data is one of the most important steps. Here, the data set has been uploaded to the google colab. Then they grouped the datasets into five different classes. Four of the classes contain various diseased leaves images. A further category was included in the dataset to separate non diseased leaves from diseased leaves.

It only includes photos of healthy leaves and diseased leaves each and every diseased leaves 35 images and healthy leaves also. Before we can extract features from the images, we need to perform some preprocessing steps to make sure that images are comparable in color, value range, and image size. The preprocessing steps are utilized from open-cv and pipelined in clustimage.

After pre-processing the images, we can start extracting features from images using the pixel value information. There are many approaches to extract features but I will focus on PCA and HOG features as these are well-established techniques that can generalize very well across different types of objects. Naturally due to limited resources only a small number of images (about 175) were collected. However, a public database was created to accept new entries to expand the database with the collaboration of users. After extracting the necessary parts from images taken, a transfer learning approach was taken to train the machine learning model. In order to find the most effective pre-trained neural network for this specific project four pre-trained networks ResNet50 [14], VGG19 [15], InceptionV3 [16] and Xception [17] were evaluated on a small dataset of images. Stochastic gradient descent, an iterative method for objectifying an optimized function [18] was used to train the neural network layers.

The absence of a substantially large dataset made it impossible to train a deep neural network from scratch. A team of researchers from Moscow took a different approach and decided to use data augmentation by only focusing on a specific part of the plant leaf images. They unfroze 39 of the 179 layers of the base neural network and alike Osokov and Goncharov (2018) they trained it using an adaptive learning optimization algorithm known as Adam optimizer [19]. This proved to be ineffective as it produced noisy data and the neural network was trained again with full size images(`432x288`). Finally, a classification accuracy of 78% was reached with a dataset of 165 images. The next approach they tried was to implement a Siamese neural taking advantage of the twin network joined by the similarity layer which calculated the distance metric between pairs of images. In this stage every image was divided into four parts which solved the problem of overfitting the neural network as there are many possible pairs of images to train on. The network was built in the same way as Simonyan, Vedaldi and Zisserman (2013) [20]. The twin networks process the images from a pair of inputs to extract a vector of necessary high-level features. With 32 filters in the initial convolutional which doubled in every layer, they finally determined their optimum architecture.

They planned to continue developing the neural network with a wider range of datasets to further improve its usefulness. Some of the drawbacks that they noticed was that since feature extraction was a primary approach to handle diseased leaf images, a lot of diseases could not be detected in primary stages. The research continued their work to develop a web-based prototype to be used in a practical scenario.

Plant disease causes substantial production and economic losses and decreases both the quality and quantity of agricultural products. That's why early detection of plant disease is an important task [21]. The traditional approach for detection and identification of plant diseases are done by humans who have expertise in relevant fields. In this paper the researcher reviewed the importance of a system that can detect the plant disease by analyzing the image of the plant leaves.

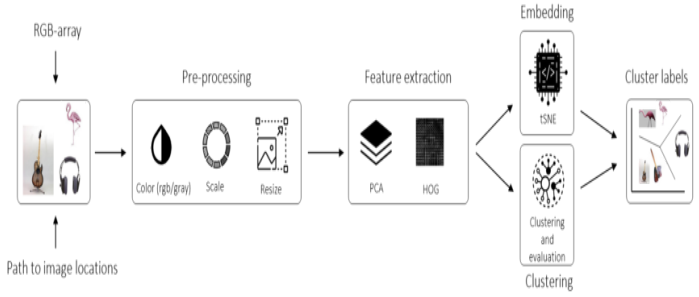
These techniques are used to analyze the healthy and diseased brinjal plant leaves. Some of the challenges that are mentioned in the paper are the effect of background data of image, and automation of the technique for continuous automated monitoring of plant diseases in real world conditions.

METHODOLOGY

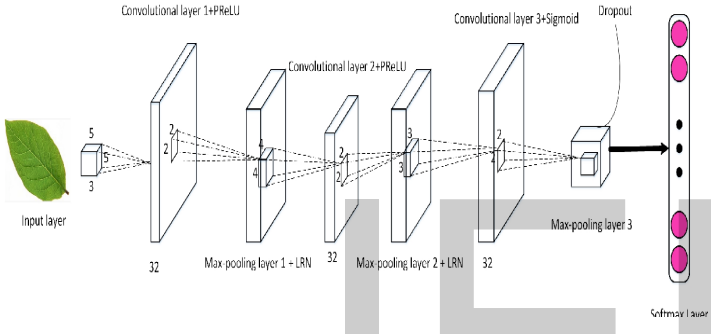
Data Acquisition & Pre-processing

Data is a central component of machine learning based projects and the collection of data is regarded as the foundation of the construction of the machine Learning model. The notion of creating a machine learning platform without a proper dataset is pointless. The better datasets we have the greater statistical model that we can build from it. To train the models we should have the right data in the right format. In this project, we've used a dataset of around hundred and seventy-five images which is published by plant Village. Plant Village is Jaffna from achveli brinjal village which is mostly interested in the thinavelli purple variety of brinjal . We are using the original dataset provided by plant Village, rather we only used the images of the brinjal (thinavelli purple) types of plant on which we're working. We removed some images and used a modified version. All images are divided into different directories. For pre-processing purposes, we have removed the noisy background colour and resized all the images. We also already pick from the brinjal plants leaves and take a photo of that leaf in front of the white sheet.

Models & Algorithms



Schematic overview of the steps taken in clustimage for unsupervised clustering of images.



Schematic overview of the steps taken in classification using convolutional neural network

Feature extraction

After preprocessing the images, we can start extracting features from images using the pixel value information. There are many approaches to extract features but I will focus on PCA and HOG features as these are well-established techniques that can generalize very well across different types of objects.

Principal component analysis (PCA)

With PCA we can reduce dimensionality and extract Principal Components (PC) where most of the variance is seen. It is utterly important that all images must have the same width and height, and also be preprocessed similarly because the pixel values will form the feature space. Suppose we have 100 grayscale 2D images of 128x128 pixels. Each image will be flattened and form a vector of size 16384. The vectors can now be stacked and form a new NxM array, where N is 100 samples and M are the 16384 features. The feature extraction will take place on the NxM array. Within *clustimage*, we can either specify the exact number of PCs to be extracted or we can set the minimum amount of explained variance that the PCs should contain in total. In the latter case, the number of PCs will be automatically determined.

Histogram of Oriented Gradients (HOG)

HOG is a feature descriptor to extract features related to the direction and orientation of edges from image data. In general, it is a simplified representation of the image that contains only the most important information, such as the number of occurrences of gradient orientation in localized portions of an image. A summary is as follows:

1. The HOG descriptor focuses on the structure or the shape of an object. HOG features contain both edge and direction information.
2. The complete image is broken down into smaller regions (localized portions) and for each region, the gradient orientation is calculated.
3. Finally, the HOG would generate a Histogram for each of these regions separately. The histograms are created using the gradient orientations of the pixel values, hence the name Histogram of Oriented Gradients.

Not all applications are useful when using HOG features as it "only" provides the outline of the image. For example, if the use-case is to group different generic objects, HOG features can do a great job but a deeper similarity within the object may be difficult as the details can be lost. Nevertheless, if an increase of HOG features is desired, you can decrease the pixels per cell.

Cluster evaluation.

At this point, we pre-processed the images, extracted the features, and with the goal in our mind, we chose the linkage type and can start clustering the images to find groups that are similar based on the distance metric. However, a clustering approach does not provide information about the separability, goodness or the optimal number of clusters. To evaluate the clustering, there are various approaches for which the Silhouette score and Davies–Bouldin (DB) are the most well-known methods. Each comes with its properties for which a summary is as follows:

- Davies-Bouldin index(dbindex): intuitively it can be described as a measure of the within cluster distances, and between cluster distances. The score is bounded between [0,1], lower is better. Note that, since it measures the distance between clusters centroids it is restricted to using the Euclidean distances.
- Silhouette score: The silhouette value is a measure of how similar a sample is to its cluster (cohesion) compared to other clusters (separation). The score is bounded between [-1,1], where a high value indicates that the object is well matched to its cluster and poorly matched to neighbouring clusters. Thus higher scores are better. In contrast to the DBindex, the Silhouette score is a sample-wise measure, i.e., measures the average similarity of the samples within a cluster and their distance to the other objects in the other clusters.

The silhouette score can be in combination with any distance metric.

For the evaluation of clusters, the clusteval library is utilized in clustimage and contains three evaluation methods: Silhouette, DBindex, and the Derivatives method. The evaluation approaches should be used in combination with clustering methods, such as agglomerative, k-means, or dbscan which are also included in clustimage. Here again, it is important to understand the mathematical properties of the methods so that it matches with the statistical properties of the (expected) clusters. As an example, DBscan in combination with the Silhouette evaluation can detect clusters with different densities and shapes while k-means assumes that clusters are convex shaped.

Be aware that the cluster evaluation approaches can easily be fooled as scores can gradually improve by an increase of the number of clusters.

Find a similar image.

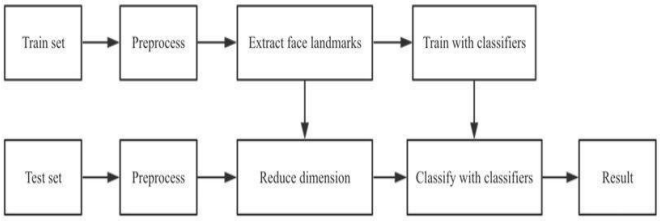
That clustering leaf dataset can be used and the clustimage library also contains a functionality to search for images. With the find functionality, an input image can be provided and similar images will be returned. Two approaches are implemented:

- Based on the K-nearest neighbor.
- Based on significance after probability density fitting.

For both approaches, the adjacency matrix is computed across all images of the specified distance metric (default Euclidean). In the case of the k-nearest neighbor approach, the k-nearest neighbors are selected, and logically it will always result in a hit. In the case of density fitting, the adjacency matrix is used to estimate the best fit for the loc/scale/arg parameters across the trained distributions [‘norm’,’expon’,’uniform’, ’gamma’, ’t’]. The fitted distribution forms the similarity distribution of samples. For each new unseen input image, the probability of similarity is computed across all images, and the images are returned that are P<alpha in the lower bound of the distribution. This approach will only return the significant hits. In the case that both k and alpha parameters are specified, the union of detected samples is taken. Let’s use the leave-dataset to find a similar image using an input image from a path location.

Convolutional neural network (CNN)

Convolutional neural networks commonly known as CNN is one of the popular deep learning algorithms for the identification and the interpretation of images. Convolutional neural network takes an image as an input and processes it. And once the necessary processing is completed, then it can classify each image under certain categories. For training and testing, each image needs to go through a series of convolutional layers with kernels, pooling and FC layers. Softmax functions are applied for classifying an object with probabilistic values of 0 to 1



CNN Workflow Diagram

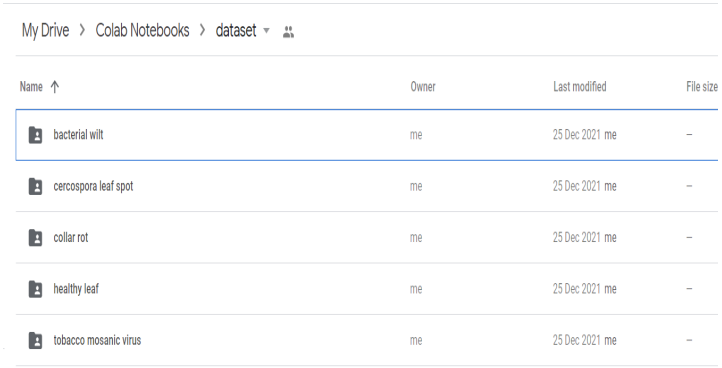
The CNN model was built mainly using the Keras library. All the images of healthy and infected plants were converted to numpy arrays using the OpenCV module. During the looping of the data it was labeled as either 1 or 0 where 1 represented ‘infected’ and 0 represented ‘healthy’ status in a plant leaf image. Images and their labels are stored in arrays from which the data is shuffled using a random module and splitted into training and testing parts. The CNN model was a simple 3 layers’ model with max pooling at the end of each layer. The accuracy was checked by removing layers. The maximum accuracy of the model was reached with 3 layers. The last layer utilized sigmoid function for binary classification.

In this classification different keras function because they increase the accuracy of those models. There are three different functions such as resnet152v2, Transfer Learning Inception V3 using Keras and Transfer Learning VGG 16 and VGG 19. These models use classify to analyze which model is better than for this classification.

ANALYSIS OF RESULTS AND OUTCOME

Data pre-processing

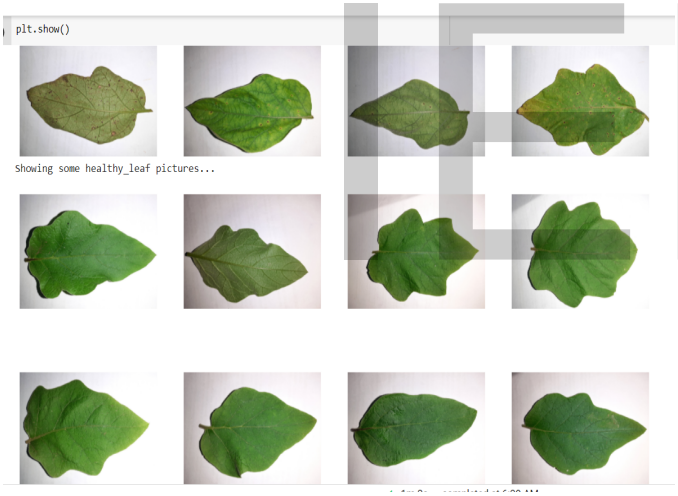
This set of code is just for illustration purposes and showing a few training images as an example. On my laptop, the data is stored in a folder one level above my Notebooks folder. Here is the organization.



Upload the images to google colab
Create directory variables, Count, and show images

```
total bacterial_wilt images: 35
total cercospora_leaf_spot images: 35
total collar_rot images: 35
total healthy_leaf images: 35
total tobacco_mosanic_virus images: 34
```

Show the count images



Show the sample images

Pre-processing & Building the object

It is the `ImageDataGenerator` means Generate batches of tensor image data with real-time data augmentation. The data will be looped over (in batches). it can be used **to augment image data with a lot of built-in pre-processing such as scaling, shifting, rotation, noise, whitening, etc**. Right now, we just use the `rescale` attribute to scale the image tensor values between 0 and 1.

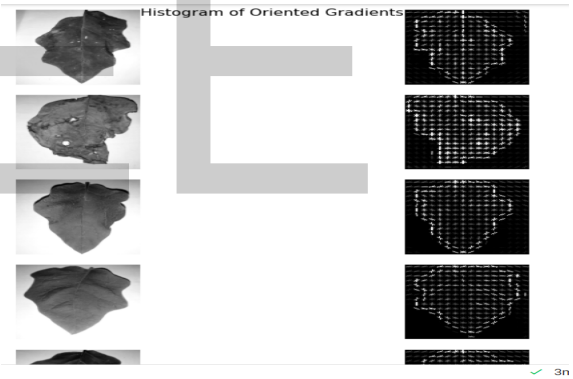
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
# All images will be rescaled by 1./255
train_datagen = ImageDataGenerator(
        rescale=1/255,
        rotation_range=40,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest')

# Flow training images in batches of 128 using train_datagen generator
training_set = train_datagen.flow_from_directory('/content/drive/MyDrive/training test data/training data',
                                                 target_size = (224, 224),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
```

```
Found 174 images belonging to 5 classes.
```

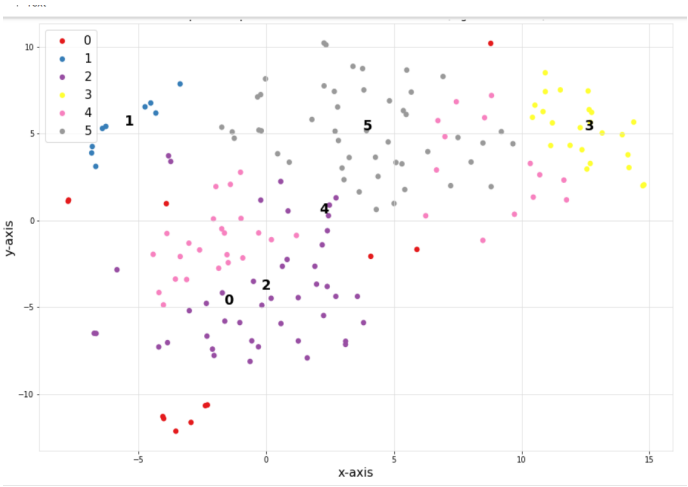
Build pre-processing with individual classes.

Extraction of image features.

After pre-processing the images, we can start extracting features from images using the pixel value information. There are many approaches to extract features but I will focus on PCA and HOG features as these are well-established techniques that can generalize very well across different types of objects.



HOG feature descriptor. Left: input image. Right: HOG features.



PCA feature descriptor.