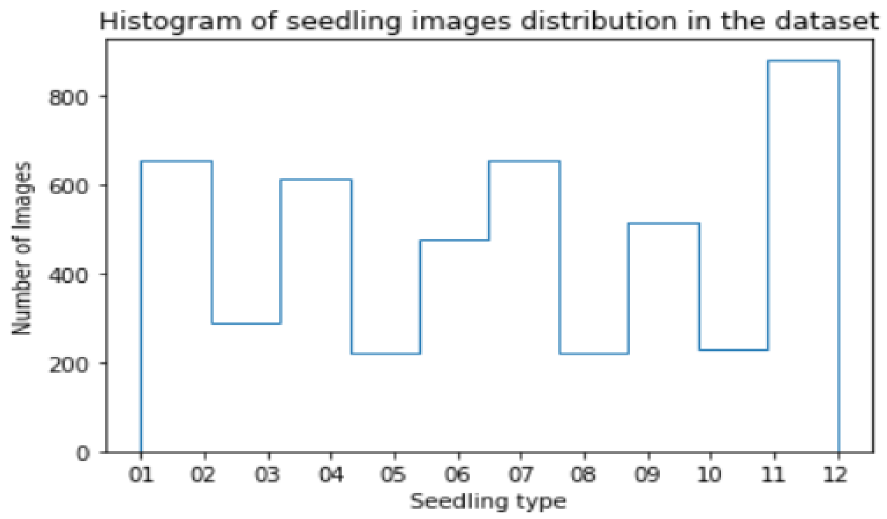
Report

1. **Personal Credentials-** Submitted by **Yashvi Gulati**, CSE- 2nd year, Bharati Vidyapeeth's College Of Engineering, New Delhi.
   Submitted to - Pucho Technology Information Private Limited

2. **Problem Statement-** Plant Seedlings Classification: Determine the species of a seedling from an image
   Link: https://www.kaggle.com/c/plant-seedlings-classification

3. **Importance-** We can do the plant seedlings classification to appropriately tell the difference between different field crops at seedling level and the difference between the seedlings and weed. This in turn allows better crop yields and allows even an inexperienced person to have better farming practices.

4. **Data-** We are provided with 4750 images of 12 different types of seedlings which are well distributed. The images are of different aspect ratios but are all above 300x300 pixels or above.



5. **Approach-** We follow a Convolutional Neural Networks based approach. A convolutional neural network (CNN) contains one or more convolutional layers, pooling layers, and fully connected layers.

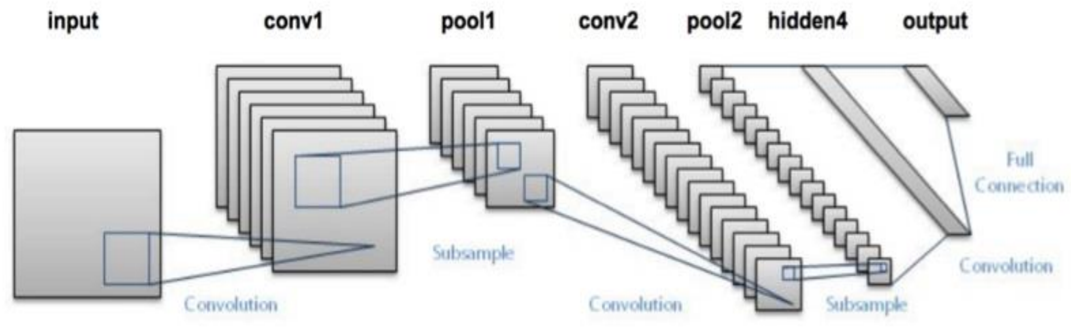
Convolution layer performs the 2 or 3 Dimensional Convolution- reversal and summation of products from shifting- with the original image to give a different set of mathematical values which represent a feature map. Further convolutional layers apply convolution operation to these feature maps. Each filter or the kernel which convolves over the image can be 2D or 3D depending on image channels and produces one feature map. Pooling layers reduce the dimensions of the feature maps for faster processing. Fully Connected layers are used for classification/regression purposes. The feature maps are first flattened and then fed to the FC layers.

6. **Code Instructions:** For running the code in your system, download the training data given at https://goo.gl/fKvdQK . Do not do data manipulations as it had already been put in a specific order which makes it easier to load in batches in the code. Change the *train_path* variable to the folder path at which you extracted the Pucho.rar. Run the rest code using Jupyter notebook (for better visual presentation) or through python file provided in any python editor-compiler.
7. **Code-** The code is mainly divided into 3 parts- Data Preprocessing, Model Generation and Training the model and checking its accuracy. We have used Python for implementing the code and Tensorflow library for numerical computations and neural network generation.
    a. **Libraries used-**
        i. Tensorflow- For making the neural network model and training and evaluating the model.
        ii. OpenCV- This is used for processing and manipulating the images in the dataset. We have mainly used the library for loading the images as numpy arrays and for resizing the image to smaller dimensions, so that we can easily run the code on our low computational laptops (CPU).
        iii. Numpy- The library is used for numerical computations and representation of our data in arrays.
    b. **Data Preprocessing-** The data is put in a folder named Train. Here, all the images are named as *label01. (1), label02. (100), ...* and so on. The number corresponding to the label represents the class of the seedling in the original data (Supervised learning problem). The number in the parentheses represents different data-points/images of that class. There are total 12 labels- *label01, label02,... label12.* These represent the corresponding 12 classes of the seedlings in the order

Black-grass
Charlock
Cleavers
Common Chickweed
Common wheat
Fat Hen
Loose Silky-bent
Maize
Scentless Mayweed
Shepherd's Purse
Small-flowered Cranesbill
Sugar beet
We first load the file path. Since we do a batch processing due to our laptop's limitations, we use the *create_batch* function. It does 3 purposes- It loads the given number (batch) of images as numpy arrays, specified as arguments to it, it assigns correct labels to the images from their name using the function *label_function*, and it processes the images by resizing them to the specified image size (*imgsize)*. These loaded batches are further fed to the network for training and evaluation purposes.

c. **Model Generation-** The model is generated by 4 main functions. Weights and Biases are non 0(with a small of noise). Also they ar positive since we use ReLU activated neurons which might be considered dead with negative weights. Both weights and biases are added as variables using tf.variable functions. Truncated.normal creates a 0 mean 0.1 standard deviation Gaussian distributed random values which is clipped after 0.2 from both sides. Convolution 2d and max pooling functions are applied to perform the convolution.

   A 2 convolutional layered network is formed. The first convolutional layer has a 5 x 5 filter with a 3 depth (due to the three channels) which convolves over the original images (say of 32 x 32 pixels) to produce 32 feature maps. These feature maps are max pooled to produce 32 feature maps of size 16 x 16 x 3. The second convolutional layer has a similar filter and produces 64 feature maps which are max pooled to the size of 8 x 8 x 3. These feature maps are then flattened and fed to a fully connected layer, fc1, having 1024 neurons which are made to learn the classification and identification. The fc1 layer is connected to the read out layer having 12 neurons, each giving out a score to classify the seedling. **The model is so chosen keeping in mind the limitations of**



the laptops we used and CPU enabled tensorflow only which prevents us to use a deeper network and bigger image sizes. This also affects the performance of our model, which could be possibly scaled to yield better results.

d. **Training the model-** The model is trained in 24 iterations and any given number of epochs. 190 batch size is taken. This means that 4560 shuffled images are taken up for training and the rest 190 images are used for validation purpose.
e. **Evaluation-** We have achieved a baseline accuracy of 40% on the training as well as the validation data. We use accuracy as the evaluation metric due to well balanced distribution of the dataset we got. The original problem on Kaggle asks for a test image submission which they have not given any verification for without submission and as this competition is now closed, we have not used the test dataset. Hence we follow only a validation approach to evaluate the model.