EXISTING METHOD

Classifying weed using a Convolutional Neural Network (CNN) involves training the model to recognize different features and characteristics of the plant that differentiate one strain from another. To classify weed using a CNN, the first step is to gather a dataset of images of different plants and weeds. The images should be labeled with the correct name (weed or crop) to train the CNN to associate the specific features of each type with its name. The next process involves separating the dataset for training and validation. The validation set is used to fine-tune the model's hyperparameters while the training set is used to train the CNN. On the basis of fresh, untested data, the CNN's performance is assessed using the testing set.

WEED CLASSIFICATION AND DETECTION IN FIELDS

For classification of weed images, the Convolutional Neural Network (CNN) is used for Feature extraction and Dense layer with Softmax activation is used for classification. The workflow of the CNN based classification is listed below.

Work Flow

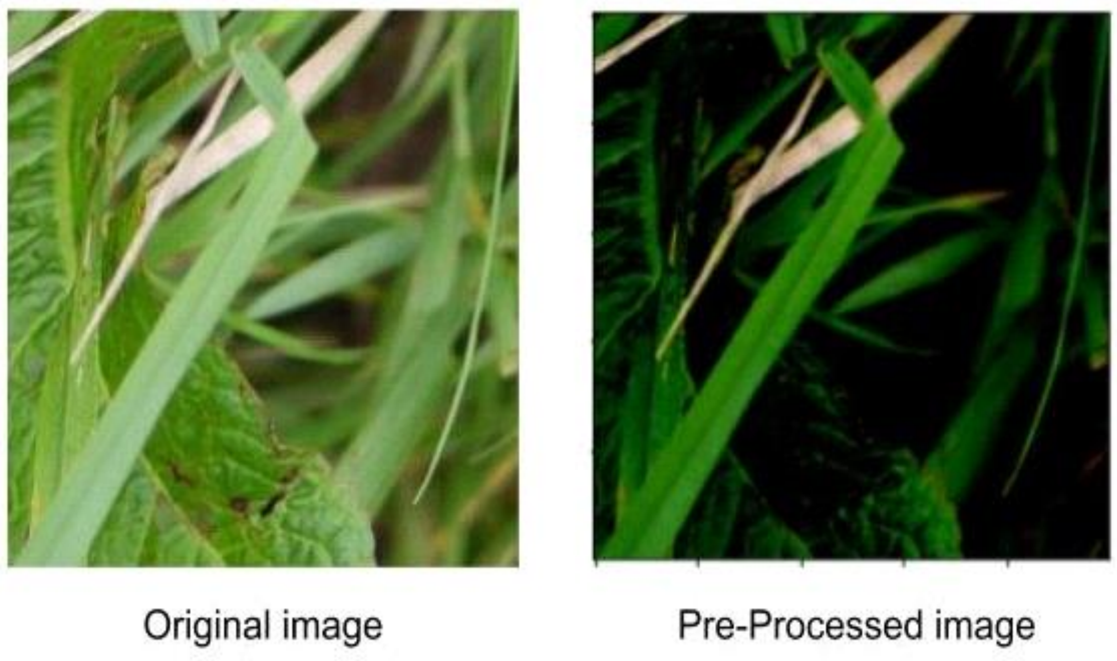
- Data pre-processing
- Data Augmentation
- Classification

PRE-PROCESSING

- The data set contains pictures of different type of crops as well as pictures of weeds in the same field that usually grows along with the crop. The dataset consists of around 2070 images which includes 1717 images were weed images and the remaining 353 images were of the crops.

- The dataset is uneven since there are significantly less samples for one class (weeds) than for the other class (crop). Data augmentation is a common approach to this issue. Using transformations like scaling, translation, rotation, and more, data augmentation is a technique for generating extra data from already-existing data.

- After treating the imbalance of data, it is fed into the CNN for the feature extraction. Further preprocessing consists of cropping/resizing and normalizing images is shown in below figure. Normalization of images helped in controlling the gradients during back propagation.



The comparison of original image vs preprocessed image

DATA AUGMENTATION

It is a method for extending the training dataset unnaturally by altering copies of a dataset made from previously collected information. It involves making little adjustments to the dataset or creating additional information through deep learning. In this, first sorting imbalance of dataset is done and then data augmentation is completed and then moving forward all the data to the loading data. In simple words it is nothing but when there are less number of data, the sample count can be increased with the help of augmentation by creating the various image version like rotating the same image into each degree of separate version of same image.

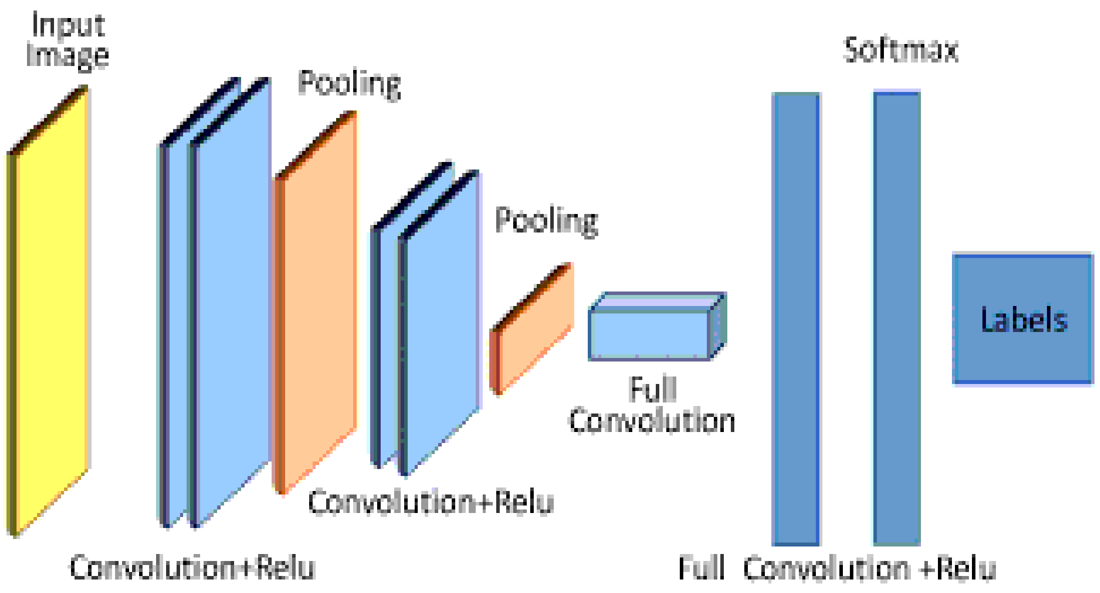
CNN ARCHITECTURE

Convolutional Neural Network are designed to work with images. The main feature of the CNN architecture is that it uses the spatial information which is the relationship between the nearby pixels. The CNN architecture is complicated when compared to other image processing architecture such as MLP. CNN consists of different additional layers and operations. Unlike other architecture which takes flattened images as input CNN takes original image as input.

Three primary layers make up CNN,

- Convolutional layers
- Pooling layers
- Fully connected layers

There are several operations involved in these layers of the CNN, which include Convolution, Pooling, Flatten and Classification operations.



Architecture Of CNN

The above figure shows the architecture of the simple CNN which consists of all the three convolutional, pooling and fully connected layers mentioned above.

Convolutional Layer

It is possible for the CNN architecture to include many layered convolutions. A convolutional layer serves as the initial layer in the CNN architecture and receives the input pictures to start the process. Convolutional layers' primary goal is to extract a collection of characteristics from a picture without affecting the connection between neighboring pixels.

The convolutional layer consists of three elements,

- Input Image
- Filters
- Feature map

Convolution

Convolution is the main operation which takes place in the convolutional layers. The convolution operation can be simply defined as the elementwise multiply-sum a process between images section and the filters.

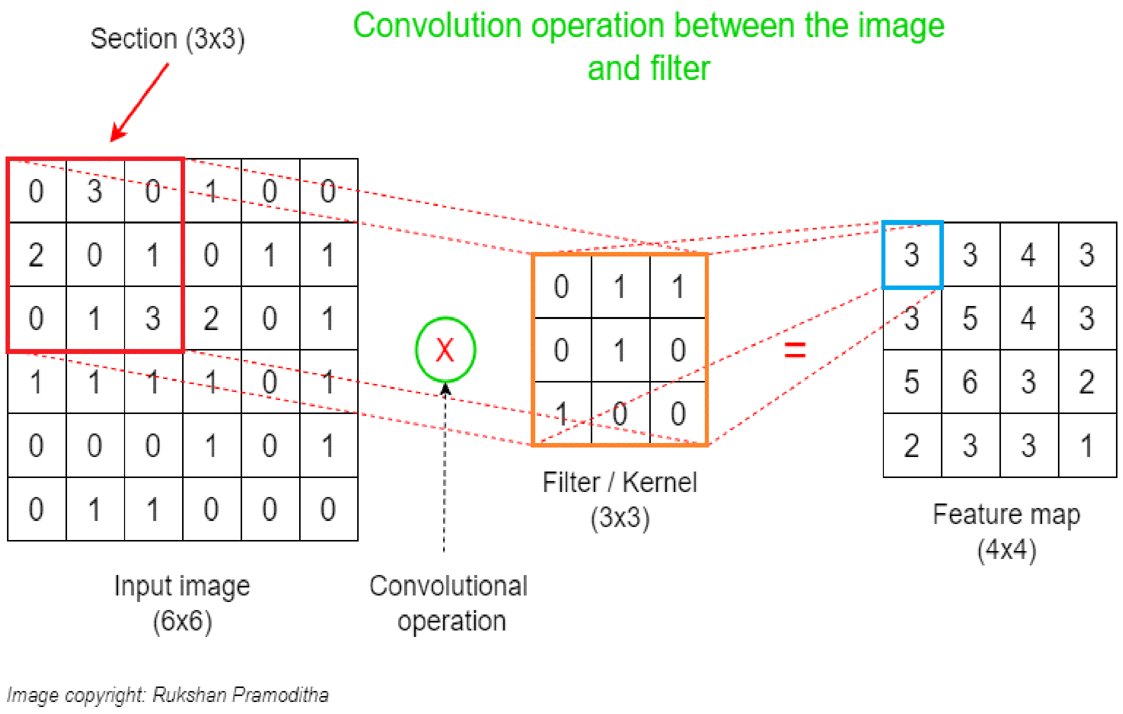


Figure Convolution operation between the image and filter

The above figure shows the result from the filter and picture's combination the Feature map as the output. The filter is the matrix which is used for feature detection. There can be multiple filters in a single convolutional layer. Different filters are used to identify different set of features from the image. The parameters like the dimension as well as the amount of filter of the filter should be given as hyperparameters. The filters are the type of parameters which are learned by the CNN while training with the dataset.

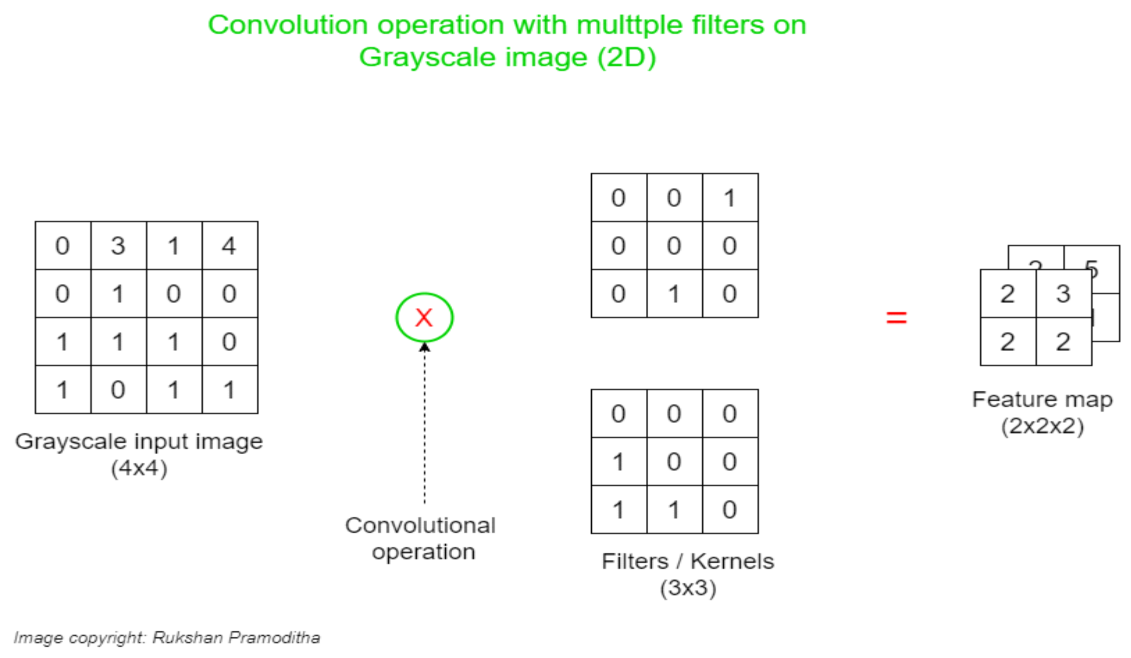
The feature map is the outcome of the convolutional layer which consist of the result of the convolution operation between the image section and the filter which will be applied across the whole image. The count of elements in the feature map equals the quantity of different image sections that is obtained by striding the filter with a user specified stride length on the image.

$$= (0*0 + 3*1 + 0*1) + (2*0 + 0*1 + 1*0) + (0*1 + 1*0 + 3*0)$$
$$= 3 + 0 + 0$$
$$= 3$$

The convolution is simply defined as the sum of the product of the terms of the image section and the filter.

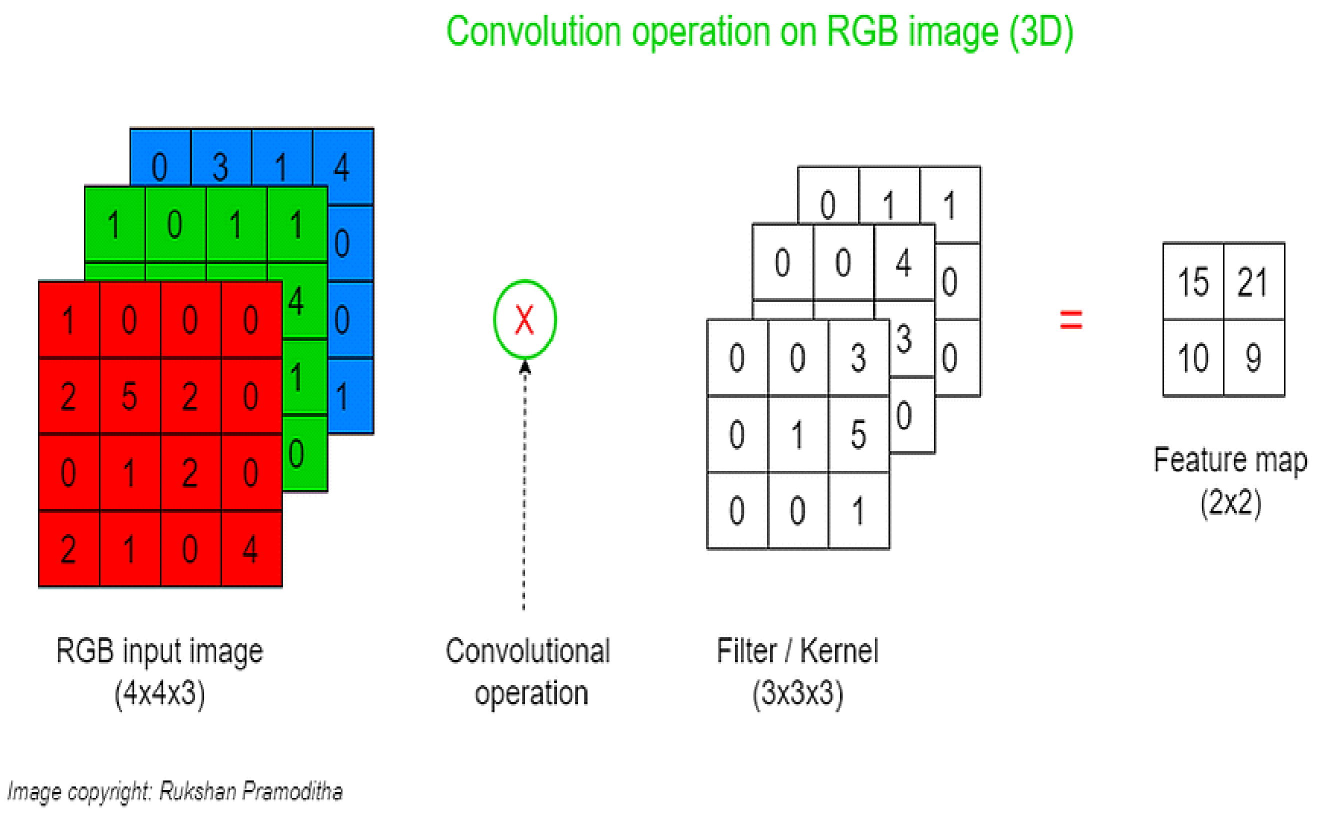
Strides and padding are introduced in the convolutional layer for the further processing. Strides are the number of steps that the filter needs to be shifted over the input image. Strides can be specified for both horizontal and vertical movement. Paddings are added to avoid reducing the size of the feature map, hence other operations can be done with the feature map.

Convolution process usually involves multiple filters. When multiple filters are used the difference in the output is that the feature map gets added with the third dimension, which denotes the number of filters. The figure 3.4 shows the convolution process with multiple filters, in which the feature map has another dimension to denote the number of filters used in the convolution process.



Convolution operation with multiple filter

When applying the convolution with the RGB image, the input image has 3 channels as shown in the below figure 3.5 one channel for each of the red, green and blue color. In this case the filter will also possess the 3 channels. The feature map is the result of convolutional layer which does not have 3 channel or additional dimension instead the results of the convolution of each channel are added up at the end to produce the feature map.



Convolution operation on RGB Image

The above Figure shows the process involved in the convolution layer between the RGB image and the filter which have 3 channels.

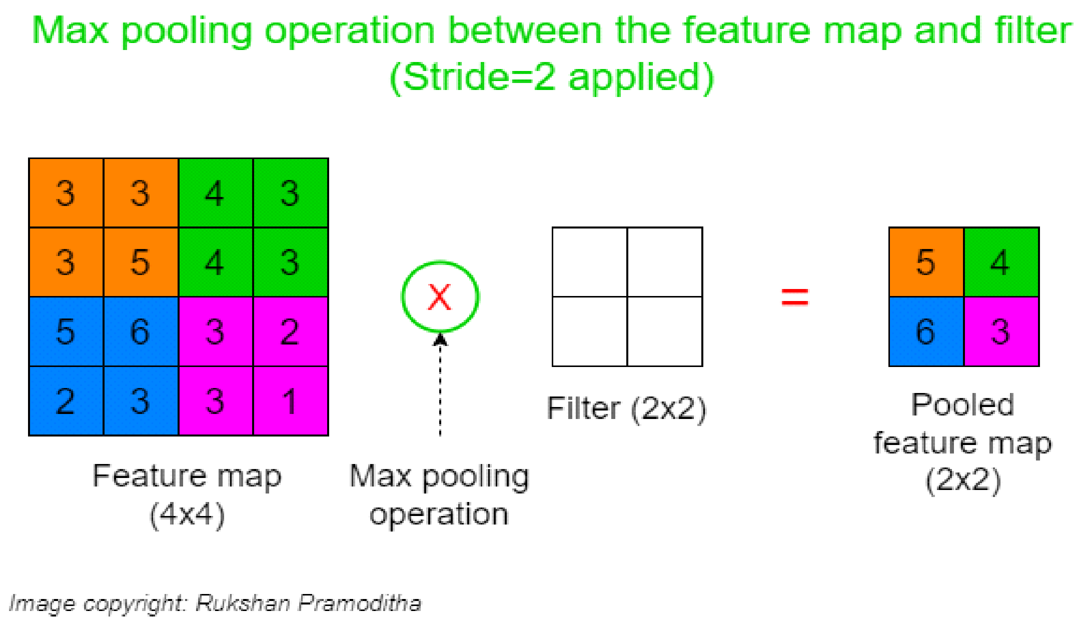
Pooling Layer

The second layer in the CNN architecture is the Pooling layers. There can be multiple pooling levels in a CNN. Each A layer for pooling comes after the transformation layer. So, convolution and pooling layers are used together as pairs. The main objective of the pooling layer is to extract the feature from the pictures and reducing the dimensions of the output returned from the previous convolutional layers.

The Pooling operation also involves three parameters, the Feature Map, the Filter and the Pooled feature map. This time, the filter is only a window because it has no elements. At the pooling layer, there are therefore no parameters to learn. Just one area of the feature map is specified using the filter. The user should specify the filter's size as a hyperparameter. The size needs to be less than the feature map's size. Use a filter with the same number of channels as the feature map if it contains multiple channels.

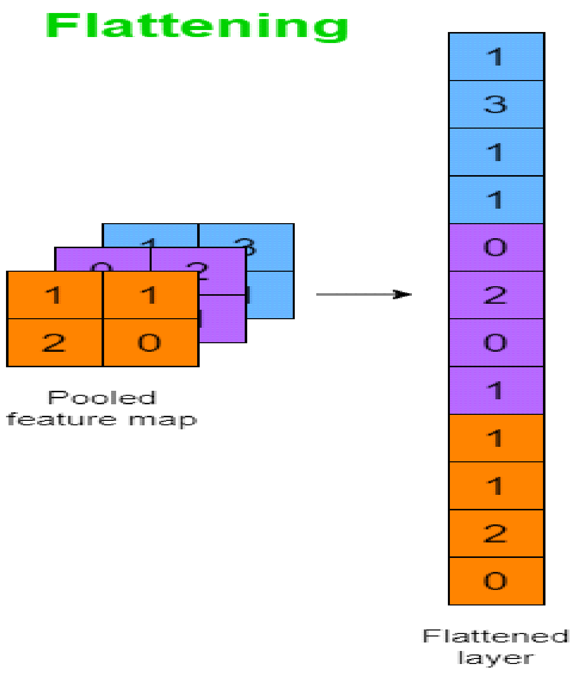
The feature map section's size ought to match the size of the filter. In the feature map, one may reposition the filter both vertically and horizontally to generate various sections.

The results of various pooling procedures between various feature map sections and the filter are stored in the pooled feature map. This will serve as the starting point for the flattening process as shown in above figures the subsequent convolution layer, if any.



Max pooling operation between the feature map and filter

The final pooled feature map output from the last pooling layer is sent to a Multilayer Perceptron (MLP) in a CNN, which may categorize the output into a class label. Data can only be one dimension for an MLP. Figure 3.7 shows the flattening process in the CNN architecture. So, the final pooled feature map must be flattened into a single column that contains the MLP's input data. Important pixel dependencies are kept when pooled maps are flattened, unlike when the original picture is flattened.



Flattening layer

Fully Connected Layers

This is the last layer in the CNN architecture. There can be many fully connected layers. The final layer does the classification (or other relevant) task. An activation function is used in each fully connected layer. The main objective of the fully connected layer is to classify the detected features in the image into a class label.

CLASSIFICATION

Feature Extraction

With convolutional neural networks, it is possible to extract features from pictures quite well. They have demonstrated superior performance over well-established feature extraction techniques as HOG, Wavelet Transform, FFT, and SIFT, to mention a few. Yet, training CNNs from scratch may be highly computationally costly. The transfer learning is employed for this purpose.

In transfer learning for image classification, a pre-trained model is used as a starting point and then fine-tuned to recognize new classes of images. This is done by training the model on a smaller dataset of images relevant to the new task and adjusting the weights of the pre-trained model. This approach allows for the creation of a high-quality image classification model for a new task without having to train a model from scratch.

During the convolutional layers, the network applies a set of learnable filters (also known as kernels) to the input image to extract features such as edges, corners, and textures. These filters slide over the entire input image, computing dot products between the filter weights and the corresponding image pixels to produce a feature map. The pooling layers then down sample the feature maps, reducing their dimensionality while retaining the most important features. This helps to reduce the computational complexity of the network while also making it more robust to small variations in the input image.

Finally, the fully connected layers take the output of the last pooling layer and flatten it into a 1D vector, which is then fed through a set of dense layers that output a probability distribution over the different classes of the problem. The class with the highest probability is then selected as the predicted class for the input image. During the training process, the network's weights are adjusted using backpropagation and gradient descent, minimizing the difference between the predicted class probabilities and the true class labels for a given set of training examples.