Assignment No.3

Aim: Build the Image classification model by dividing the model into the following four stages:

a. Loading and preprocessing the image data

b. Defining the model's architecture

c. Training the model

d. Estimating the model's performance

Objective: Student will learn:

1. To be able to apply deep learning algorithms to solve problems of moderate complexity
2. Understand how a model is trained and evaluated.
3. Classifying images from the image dataset.
4. Our main goal is to train a neural network (using Keras) to obtain > 90% accuracy on image dataset.. 5. To apply the algorithms to a real-world problem, optimize the models learned and report on the expected accuracy that can be achieved by applying the models.

Methodology to be used -

1. Deep Learning
2. Convolutional Neural Network

Theory:

Deep Learning has been proved that it's a very powerful tool due to its ability to handle huge amounts of data. The use of hidden layers exceeds traditional techniques, especially for pattern recognition. One of the most popular Deep Neural Networks is Convolutional Neural Networks (CNN).

Convolutional Neural Networks (CNNs):

A convolutional neural network (CNN) is a type of Artificial Neural Network (ANN) used in image recognition and processing which is specially designed for processing data (pixels). The goal of a CNN is to learn higher-order features in the data via convolutions. They are well suited to object recognition with images and consistently top image classification competitions. They can identify faces, individuals, street signs, platypuses, and many other aspects of visual data. CNNs overlap with text analysis via optical character recognition, but they are also useful when analyzing word 6 as discrete textual units. They're also good at analyzing sound. The efficacy of CNNs in image recognition is one of the main reasons why the world recognizes the power of deep learning. CNNs are good at building position and (somewhat) rotation invariant features from raw image data. CNNs are powering major advances in machine vision, which has obvious applications for self-driving cars, robotics, drones, and treatments for the visually impaired. The structure of image data allows us to change the architecture of a neural network in a way that we can take advantage of this structure. With CNNs, we can arrange the neurons in a three-dimensional structure for which we have the following:Width Height Depth

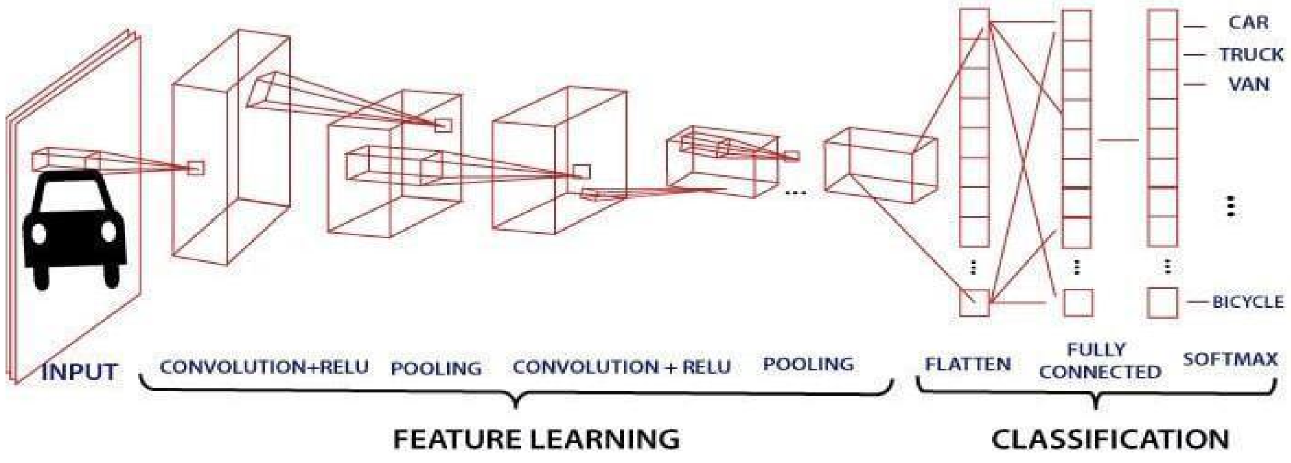
These attributes of the input match up to an image structure for which we have:

Image width in pixels ,Image height in pixels ,RGB channels as the depth

We can consider this structure to be a three-dimensional volume of neurons. A significant aspect to how CNNs evolved from previous feed-forward variants is how they achieved computational efficiency with new layer types.

CNN Architecture Overview:

CNNs transform the input data from the input layer through all connected layers into a set of class scores given by the output layer. There are many variations of the CNN architecture, but they are based on the pattern of layers, as demonstrated in Figure 1 (below)



Convolution Layer:

Convolution layer is the first layer to extract features from an input image. By learning image features using a small square of input data, the convolutional layer preserves the relationship between pixels. It is a mathematical operation which takes two inputs such as image matrix and a kernel or filter. o The dimension of the image matrix is **h×w×d**.

o The dimension of the filter is $\mathbf{f_h \times f_w \times d}$.

o The dimension of the output is $\mathbf{(h-f_h+1)\times(w-f_w+1)\times 1}$.

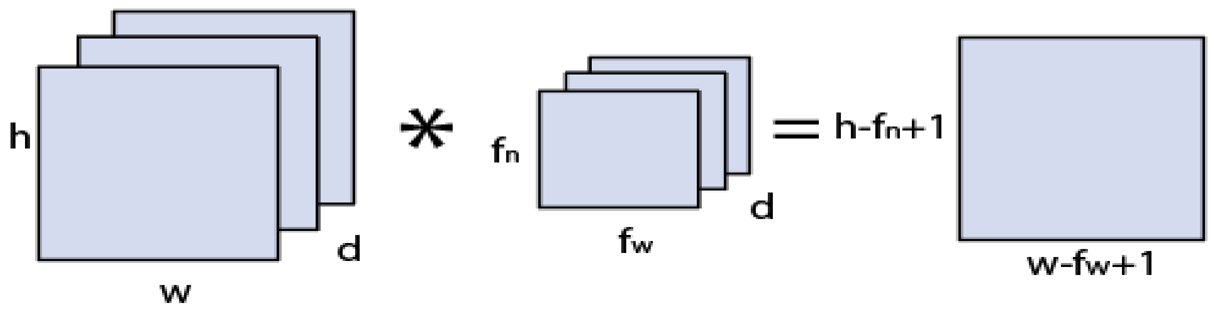
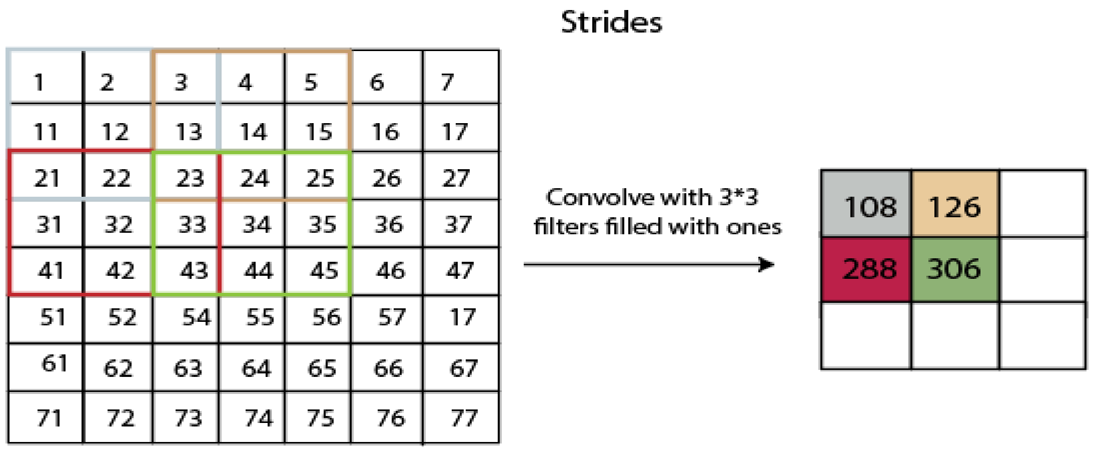


Image matrix multiplies kernl or filter matrix

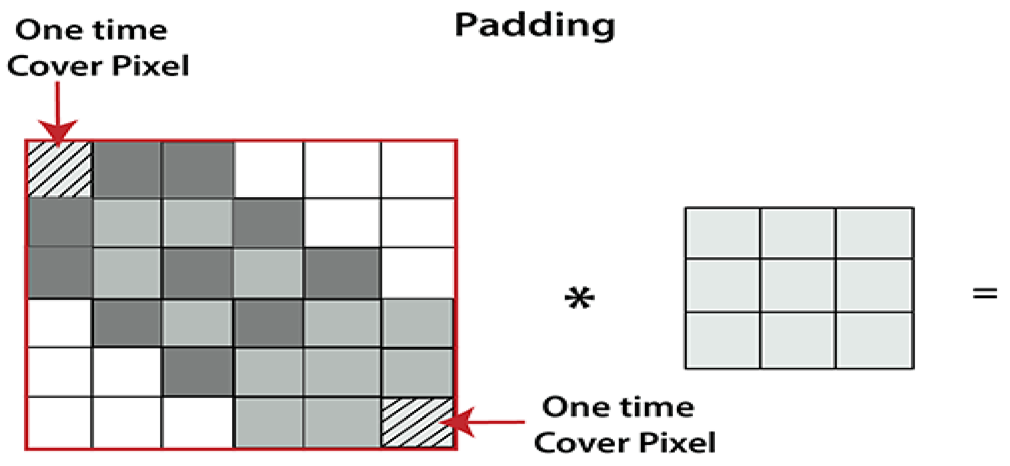
Strides:

Stride is the number of pixels which are shifted over the input matrix. When the stride is equal to 1, then we move the filters to 1 pixel at a time and similarly, if the stride is equal to 2, then we move the filters to 2 pixels at a time. The following figure shows that the convolution would work with a stride of 2.



Padding:

Padding plays a crucial role in building the convolutional neural network. If the image shrinks and if we take a neural network with 100's of layers on it, it will give us a small image after it is filtered in the end. If we take a three by three filter on top of a grayscale image and do the convolving then what will happen?



It is clear from the above picture that the pixel in the corner will only get covers one time, but the middle pixel will get covered more than once. It means that we have more information on that middle pixel, so there are two downsides:

- o Shrinking outputs
- o Losing information on the corner of the image.

To overcome this, we have introduced padding to an image. **"Padding is an additional layer which can add to the border of an image."**

Pooling Layer:

Pooling layer plays an important role in pre-processing an image. Pooling layer reduces the number of parameters when the images are too large. Pooling is "**downscaling**" of the image obtained from the previous layers. It can be compared to shrinking an image to reduce its pixel density. Spatial pooling is also called downsampling or subsampling, which reduces the dimensionality of each map but retains the important information. There are the following types of spatial pooling:

Max Pooling:

Max pooling is a **sample-based discretization process**. Its main objective is to downscale an input representation, reducing its dimensionality and allowing for the assumption to be made about features contained in the sub-region binned.

Max pooling is done by applying a max filter to non-overlapping sub-regions of the initial representation.

Average Pooling:

Down-scaling will perform through average pooling by dividing the input into rectangular pooling regions and computing the average values of each region.

Syntax

layer = averagePooling2dLayer(poolSize)

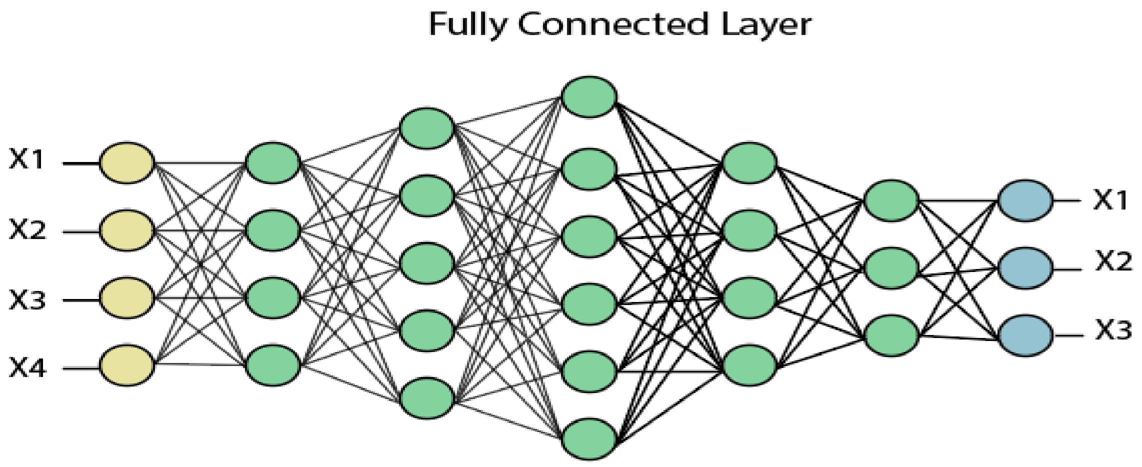
layer = averagePooling2dLayer(poolSize,Name,Value)

Sum Pooling:

The sub-region for **sum pooling** or **mean pooling** are set exactly the same as for **max-pooling** but instead of using the max function we use sum or mean.

Fully Connected Layer:

The fully connected layer is a layer in which the input from the other layers will be flattened into a vector and sent. It will transform the output into the desired number of classes by the network.



In the above diagram, the feature map matrix will be converted into the vector such as **x1, x2, x3... xn** with the help of fully connected layers. We will combine features to create a model and apply the activation function such as **softmax** or **sigmoid** to classify the outputs as a car, dog, truck, etc.

TensorFlow is an open-source platform for machine learning. Keras is the high-level application programming interface (API) of TensorFlow. Using Keras, we can rapidly develop a prototype system and test it out. This is the first in a three-part series on using TensorFlow for supervised classification tasks.

To implement Convolutional Neural Network for image classification

CNN -

Now imagine there is an image of a bird, and you want to identify it whether it is really a bird or something other. The first thing you should do is feed the pixels of the image in the form of arrays to the input layer of the neural network (MLP networks used to classify such things). The hidden layers carry Feature Extraction by performing various calculations and operations. There are multiple hidden layers like the convolution, the ReLU, and the pooling layer that performs feature extraction from your image. So finally, there is a fully connected layer that you can see which identifies the exact object in the image. You can understand very easily from the following figure:

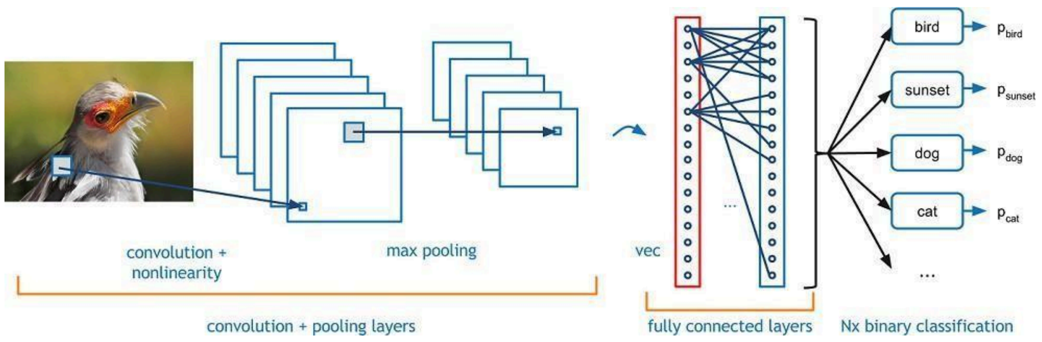


fig:Convolution and pooling layers in CNN

Convolution:

Convolution Operation involves matrix arithmetic operations and every image is represented in the form of an array of values(pixels). Let us understand example: a = [2,5,8,4,7,9] b = [1,2,3] In Convolution Operation, the arrays are multiplied one by one element-wise, and the product is grouped or summed to create a new array that represents a*b. The first three elements of matrix a are now multiplied by the elements of matrix b. The product is summed to get the result and stored in a new array of a*b. This process remains continuous until the operation gets completed.fig: Sequence of Convolution and pooling layers in CNN.

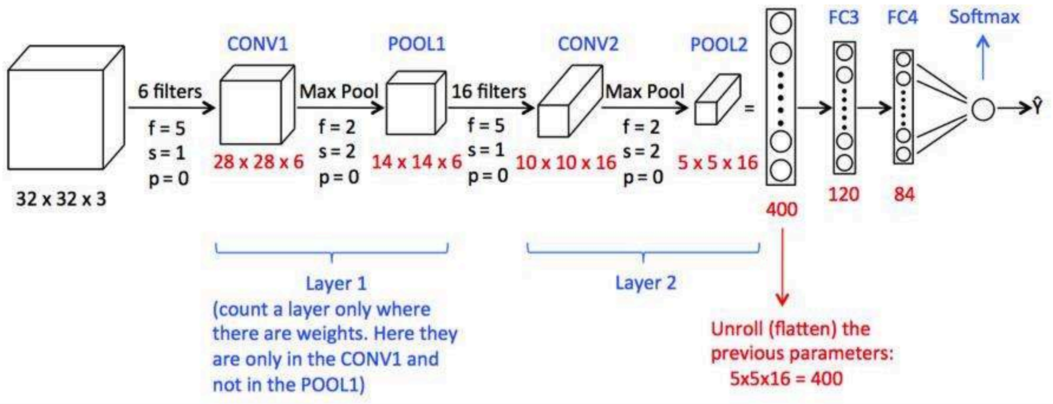
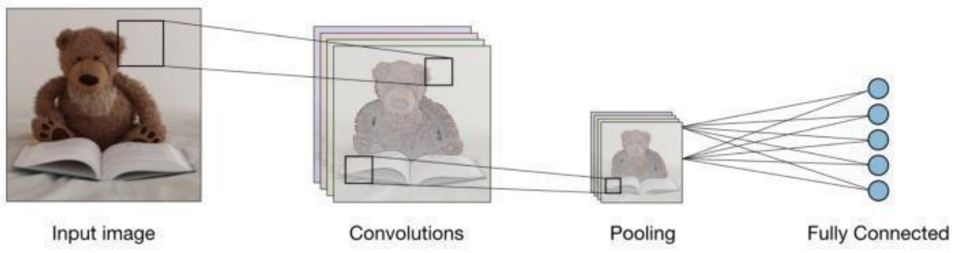


Fig-Sequence of Convolution and pooling layers in CNN

Pooling:

After the convolution, there is another operation called pooling. So, in the chain, convolution and pooling are applied sequentially on the data in the interest of extracting some features from the data. After the sequential convolutional and pooling layers, the data is flattened into a feed-forward neural network which is also called a Multi-Layer Perceptron.



Conclusion: Thus, we have implemented the Image classification model using CNN. With the above code we can see that sufficient accuracy has been met. Throughout the epochs, our model accuracy increases and loss decreases, which is good since our model gains confidence with our prediction This indicates the model is trained in a good way .