

Image Classification

CNN

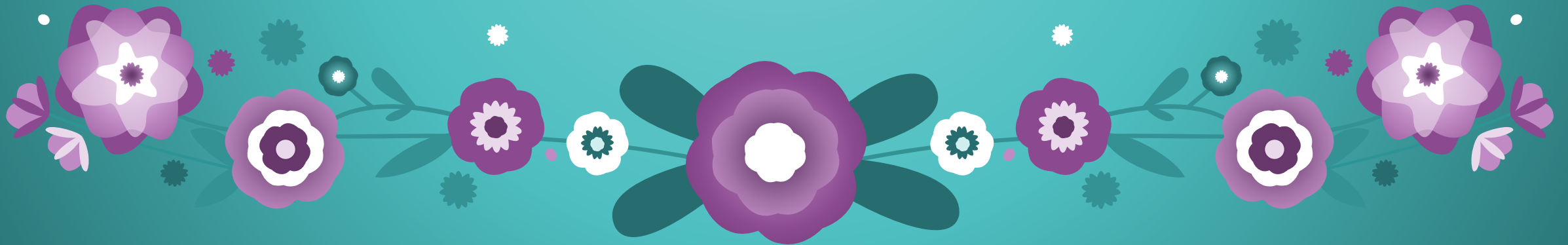




Image Composition

An image consists of the smallest indivisible segments called pixels and every pixel has a strength often known as the pixel intensity.

Whenever we study a digital image, it usually comes with three color channels, i.e. the Red-Green-Blue channels, popularly known as the “RGB” values.

Why RGB? Because it has been seen that a combination of these three can produce all possible color palettes.

Whenever we work with a color image, the image is made up of multiple pixels with every pixel consisting of three different values for the RGB channels.



RGB

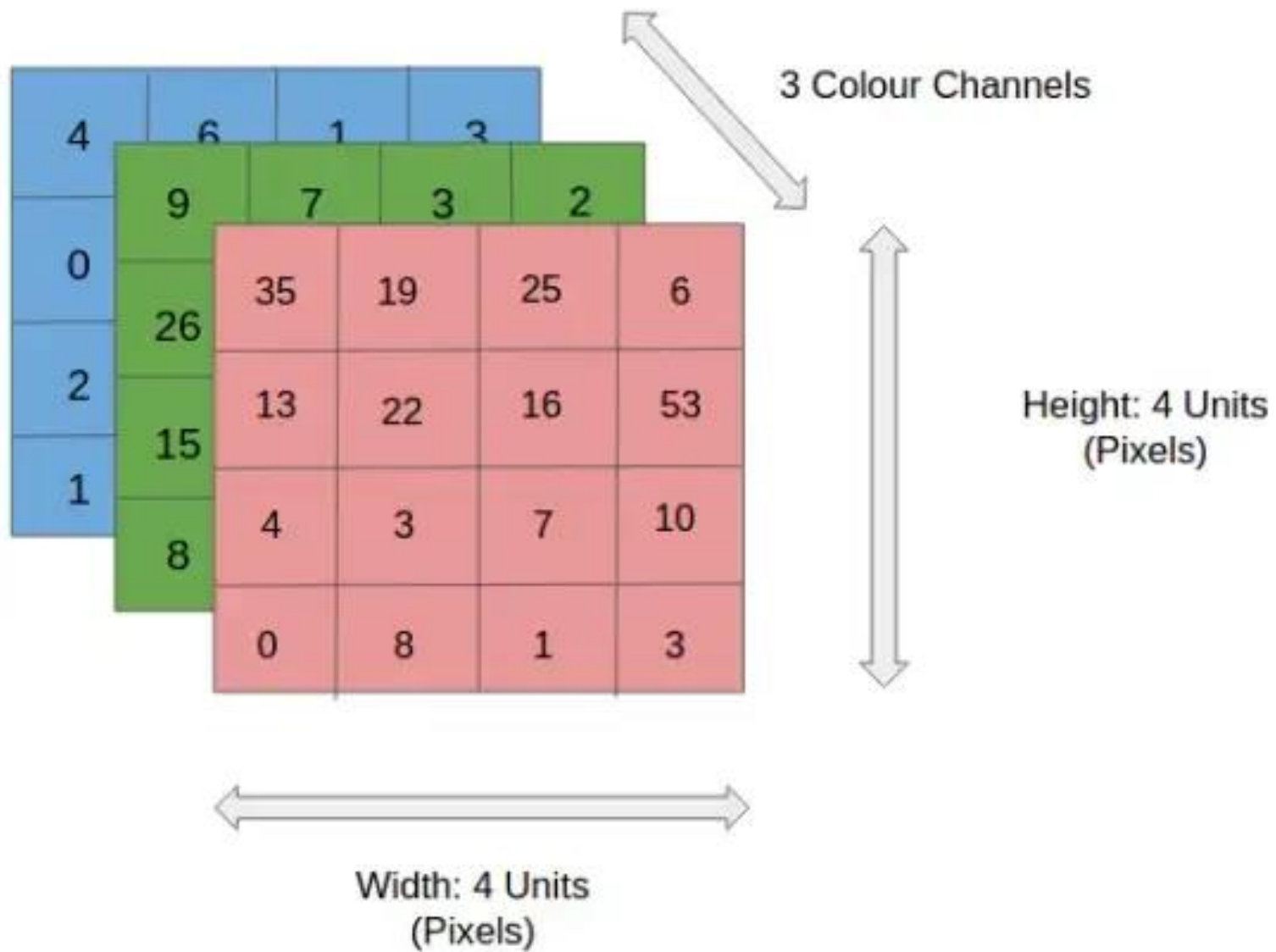


Image Normalization

Image normalization is a typical process in image processing that changes the range of pixel intensity values. Its normal purpose is to convert an input image into a range of pixel values that are more familiar or normal to the senses, hence the term normalization.

$$\text{Output_channel} = 255 * (\text{Input_channel} - \text{min}) / (\text{max} - \text{min})$$

If we are using a grayscale image, we only need to normalize using one channel. However, if we are normalizing a RGB (3 channels) we need to normalize for each channel using the same criteria.

CNN or the convolutional neural network

Is a class of **deep learning neural networks**. In short think of CNN as a machine learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other.

CNN works by extracting features from the images. Any CNN consists of the following:


- The input layer which is a grayscale image
- The Output layer which is a binary or multi-class labels
- Hidden layers consisting of convolution layers, ReLU (rectified linear unit) layers, the pooling layers, and a fully connected Neural Network



Convolution Layer

The convolution layer consists of one or more Kernels with different weights that are used to extract features from the input image. The convolution layer is the core building block of the CNN. It carries the main portion of the network's computational load.

This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but is more in-depth.



If we have an input of size $W \times W \times D$ and D_{out} number of kernels with a spatial size of F with stride S and amount of padding P , then the size of output volume can be determined by the following formula:

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

The result of this operation is a feature map that basically detects features from the images rather than looking into every single pixel value.

1 _{x1}	1 _{x0}	1 _{x1}	0	0
0 _{x0}	1 _{x1}	1 _{x0}	1	0
0 _{x1}	0 _{x0}	1 _{x1}	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

Convolved
Feature

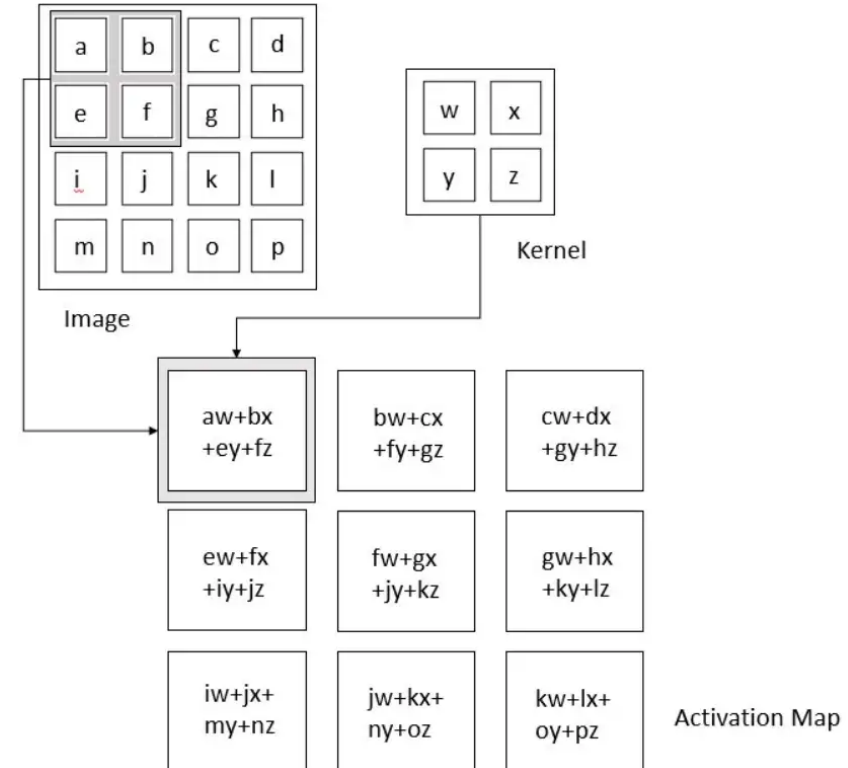
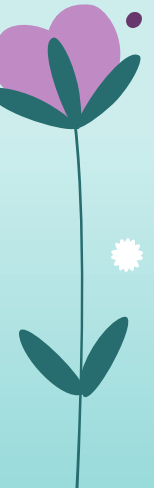
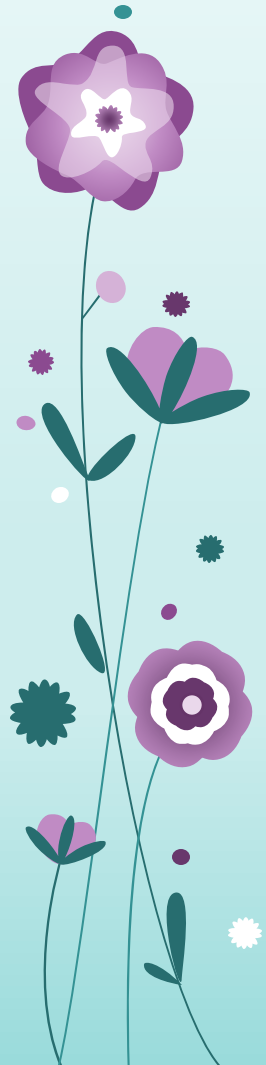


Image features, such as edges and interest points, provide rich information on the image content. They correspond to local regions in the image and are fundamental in many applications in image analysis: recognition, matching, reconstruction, etc. Image features yield two different types of problem: the detection of the area of interest in the image, typically contours, and the description of local regions in the image, typically for matching in different images.

Pooling Layer

The pooling layer applies a non-linear down-sampling on the convolved feature often referred to as the activation maps. This is mainly to reduce the computational complexity required to process the huge volume of data linked to an image. Pooling is not compulsory and is often avoided. Usually, there are two types of pooling, Max Pooling, that returns the maximum value from the portion of the image covered by the Pooling Kernel and the Average Pooling that averages the values covered by a Pooling Kernel.





The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

Max Pooling

29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

2 x 2
pool size

100	184
12	45

Average Pooling

31	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

2 x 2
pool size

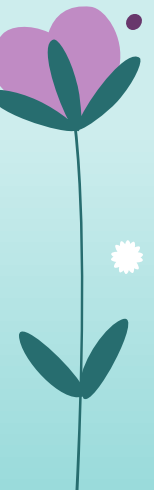
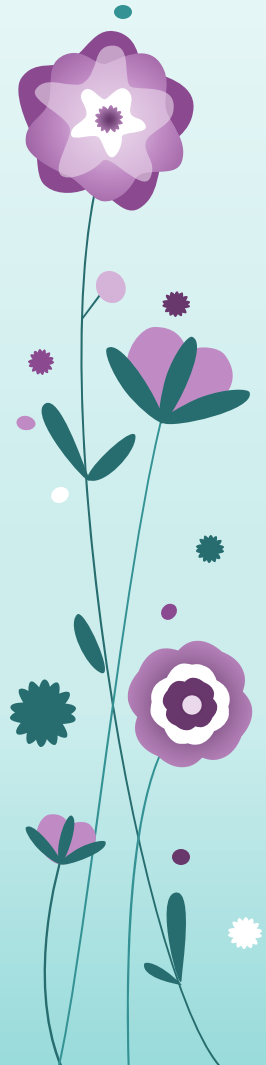
36	80
12	15

If we have an activation map of size $W \times W \times D$, a pooling kernel of spatial size F , and stride S , then the size of output volume can be determined by the following formula:

$$W_{out} = \frac{W - F}{S} + 1$$

Image Flattening

Once the pooling is done the output needs to be converted to a tabular structure that can be used by an artificial neural network to perform the classification. Note the number of the dense layer as well as the number of neurons can vary depending on the problem statement. Also often a drop out layer is added to prevent overfitting of the algorithm. Dropouts ignore few of the activation maps while training the data however use all activation maps during the testing phase. It prevents overfitting by reducing the correlation between neurons.





Fully Connected Layer

Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular FCNN. This is why it can be computed as usual by a matrix multiplication followed by a bias effect.

The FC layer helps to map the representation between the input and the output.

Non-Linearity Layers

Linear filtering is the filtering method in which the value of output pixel is linear combinations of the neighbouring input pixels. it can be done with convolution. For example, mean/average filters or Gaussian filtering.

A **non-linear filtering** is one that cannot be done with convolution or Fourier multiplication. A sliding median filter is a simple example of a non-linear filter.

Since convolution is a linear operation and images are far from linear, non-linearity layers are often placed directly after the convolutional layer to introduce non-linearity to the activation map.

ReLU

The Rectified Linear Unit (ReLU) has become very popular in the last few years. It computes the function $f(\kappa) = \max(0, \kappa)$. In other words, the activation is simply threshold at zero.

In comparison to sigmoid and tanh, ReLU is more reliable and accelerates the convergence by six times.

Unfortunately, a con is that ReLU can be fragile during training. A large gradient flowing through it can update it in such a way that the neuron will never get further updated. However, we can work with this by setting a proper learning rate.

References

- <https://towardsdatascience.com/convolutional-neural-network-for-image-processing-using-keras-dc3429056306>
- <https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939>
- <http://dev.ipol.im/~nmonzon/Normalization.pdf>

