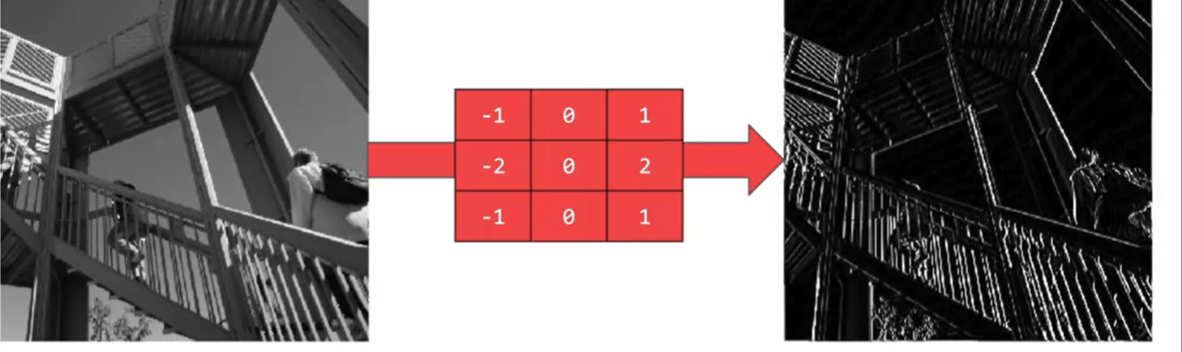
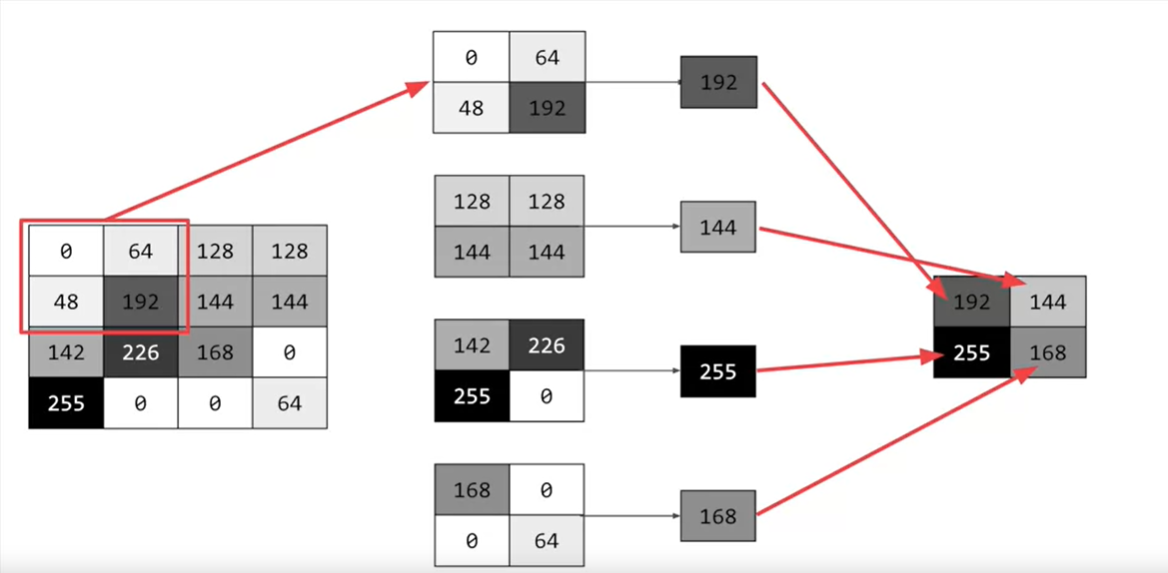
Convolutional Neural Networks CNN

Before passing the data ( Images ) into our model (and after the normalization step ) , We should add two layers before going through The Flatten and the Dense One

## Convolution Layer ( Conv2D ) :

* + It applies a set of filter to our images in order to extract new features from them in order to have a better distinguishing between the classes by adding new parameters to our DL pipeline
  + A filter is defined by a shape ( 3x3 in this example ) where the CURRENT\_PIXEL\_VALUE is the pixel positioned in the center of the filter array , and Then we multiply its neighbors by the corresponding coefficient in the filter array definition
    - Example of a particular filter which extract features from the horizontal lines :
* It’s important to note that the filter will be applied only to the intern pixels ( who have a neighbor pixels in all the sides ) ,so the pixels in the border will be removed ( so from 26\*26 we will go to 24\*24 )

## Pooling layer

* + The role of these layers is to minimize the image size ( number of pixels ) with a minimal loss of the image feature in order to make the process of the model fitting quicker ( since we are manipulating a litter images )
    - This layer select the pixel with the higher value among the 4x4 matrix , the image went up from 8x8 to 2x2

## Final Architecture:

Model = tf.keras.models.Sequential([

tf.keras.conv2D(64 ,( 3,3) , activation=”relu” , input-shape=(24,24,1)) (1),

tf.keras.MaxPooling2D(2,2) (2),

tf.keras.conv2D(64 ,( 3,3) , activation=”relu” )

tf.keras.MaxPooling2D(2,2) ,

tf.keras.layers.Flatten(),

tf.keras.layers.Dense(128, activation=tf.nn.relu,

tf.keras.layers.Dense(10, activation=tf.nn.softmax)

])

1. :
   1. - 64 : the number of the applied filters ( so it ha a the number of the generated images )
   2. (3,3) : the matrix dimension of the applied filters
   3. Input\_shape=(24,24,1) :
      1. 24,24 : the size of the image
      2. 1 : the number of channels :
         1. 1 mean a gray scale images
         2. 3 for the rgb images
         3. 4 for rgba images
2. MaxPooling is the pool layer who takes the biggest value of the pixel
   1. 2\*2 the size of the matrix where we will do our pooling ( 2\*2 like the image above )

* We repeated Conv2d and MaxPooling2D again so that the network can learn another set of convolutions on top of the existing one, and then again, pool to reduce the size