OpenCV’s blobFromImage working

Common preprocessing tasks:

1. Mean subtraction
2. Scaling by some factor

blobFromImage performs:

1. Mean subtraction
2. Scaling
3. Channel swapping (optional)

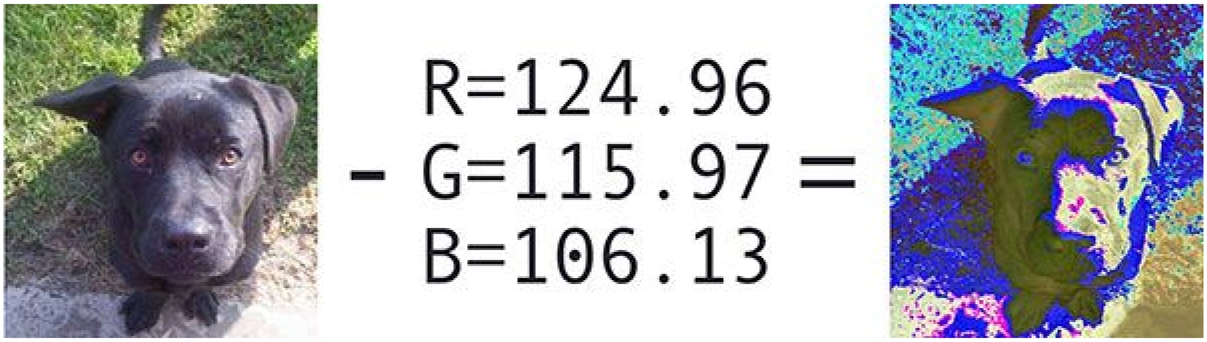
Mean subtraction:

- Used to help combat illumination changes in the input images
- hence , used to aid CNN

Before training, calculate the *average pixel intensity* across all the images in the training set for Red, Green and Blue.

[However, in some cases the mean Red, Green, and Blue values may be computed channel-wise rather than pixel-wise, resulting in an MxN matrix. In this case the MxN matrix for each channel is then subtracted from the input image during training/testing.]

When we pass our images to our network for training or testing purpose, we subtract the mean from the images.



We may also have a *scaling factor*, σ , which adds in a normalization:

$$R = (R - \mu_R)/\sigma$$

$$G = (G - \mu_G)/\sigma$$

$$B = (B - \mu_B)/\sigma$$

The value of sigma can be the std. deviation of the training set pr can be manually set by the user. It depends on the network architecture of the model.

[not all deep learning architectures perform mean subtraction and scaling!]

When $\sigma = 1$, no scaling takes place.

blobFromImage creates a 4-dimensional blob from the image. Optionally resizes and crops image from center, subtract mean values, scale values by scalefactor, swap Blue and Red channels.

[Informally, a blob is just a (potentially collection) of image(s) with the same spatial dimensions (i.e., width and height), same depth (number of channels), that have all been preprocessed in the same manner.]

blob = cv2.dnn.blobFromImage(image, scalefactor=1.0, size, mean, swapRB=True)

function returns a blob which is our input image after mean subtraction, normalizing, and channel swapping.

For processing multiple images, we may use *blobFromImages,* as it reduces the number of function calls.