Xmas Research

Maria Shevchuk

Applying Gaussian Low Pass & High Pass filters

```
from scipy.ndimage import gaussian_filter

X_train = np.array([gaussian_filter(image, sigma = 1) for image in X_train])
X_test = np.array([gaussian_filter(image, sigma = 1) for image in X_test])
```

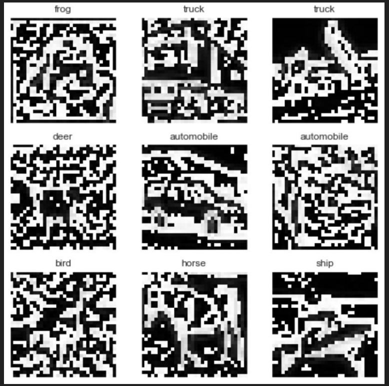
```
X_train = np.array([image - gaussian_filter(image, sigma = 1) for image in X_train])
X_test = np.array([image - gaussian_filter(image, sigma = 1) for image in X_test])
```

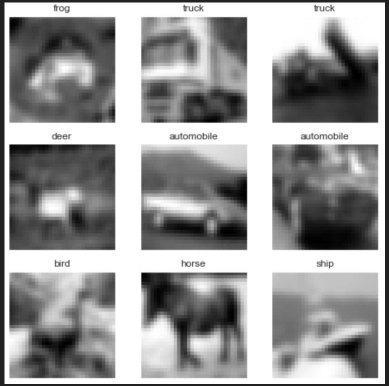
- Takes average over the nearby pixels (sigma = # of contributing pixels)

High Pass

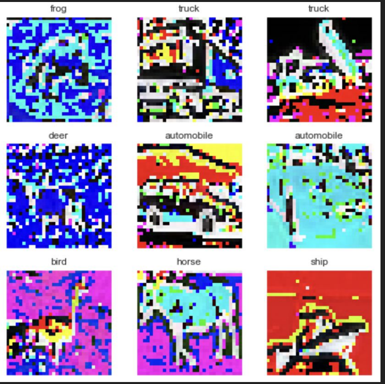
Low Pass

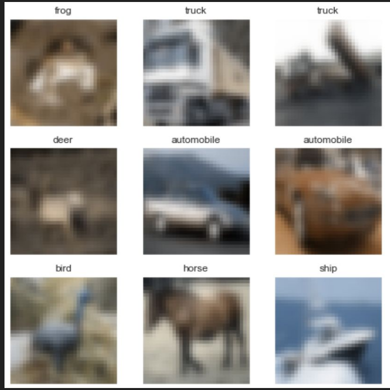
Grey





RGB

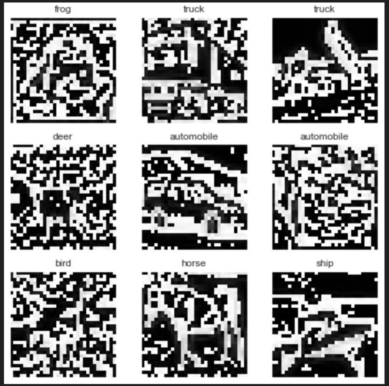


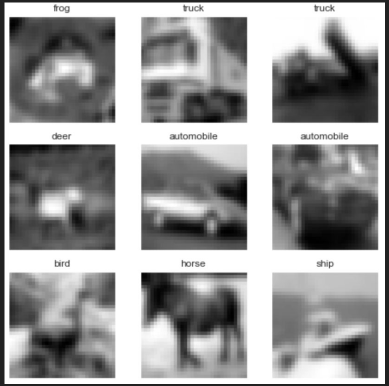


High Pass

Low Pass

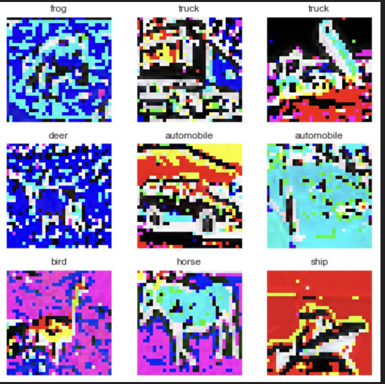
Grey

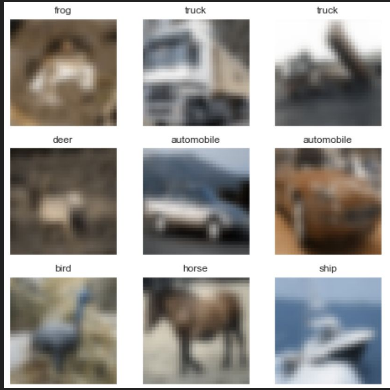




RGB

?





Averaging | cv.blur

```
X_train = np.array([cv2.blur(image, (3,3)) for image in X_train])
X_test = np.array([cv2.blur(image, (3,3)) for image in X_test])
```

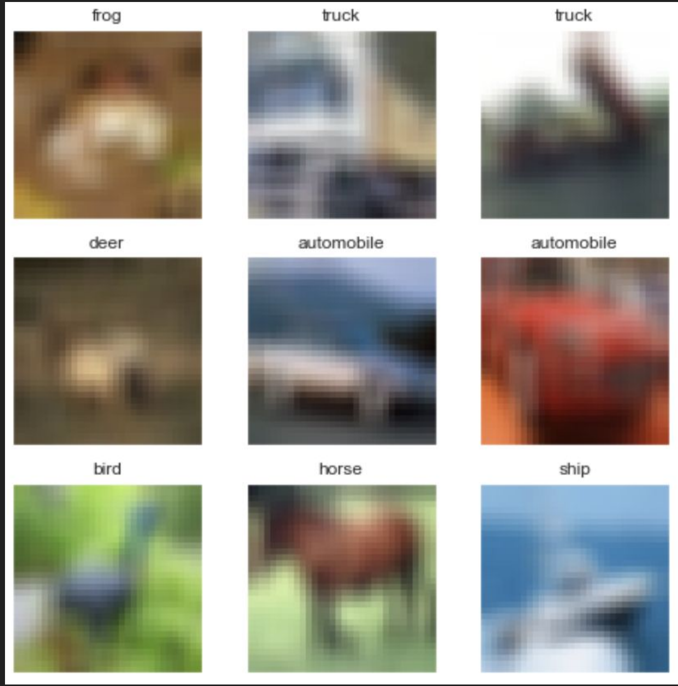
```
X_train = np.array([image - cv2.blur(image, (3,3)) for image in X_train])
X_test = np.array([image - cv2.blur(image, (3,3)) for image in X_test])
```

- Convolutes an image with a normalized box filter of specified size

$$K = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

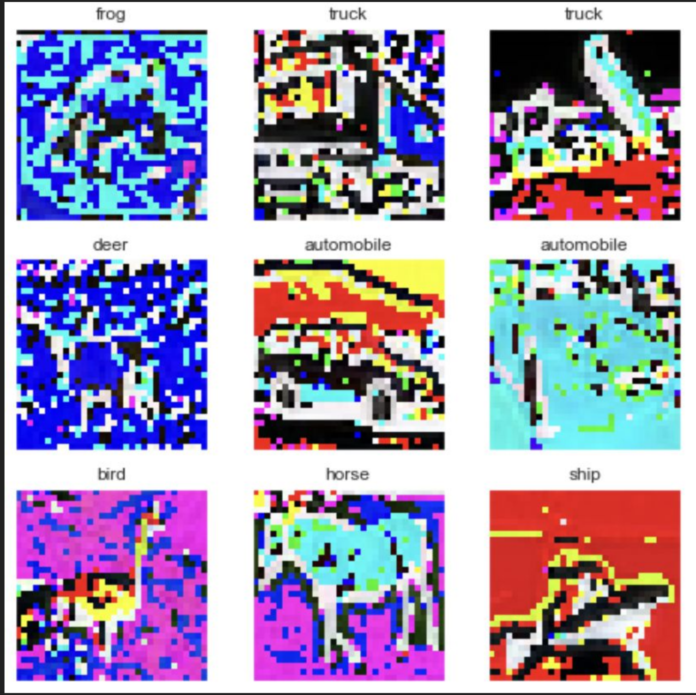
gaussian_filter

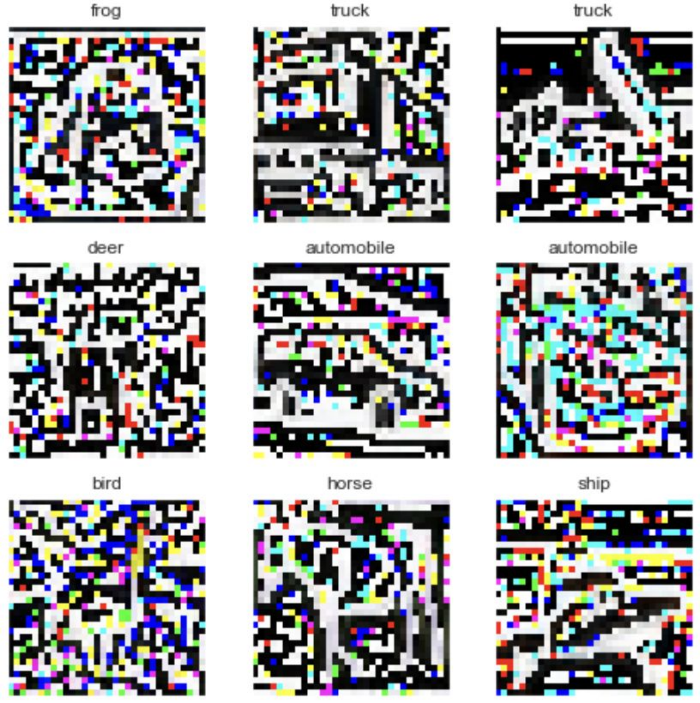
cv2.blur



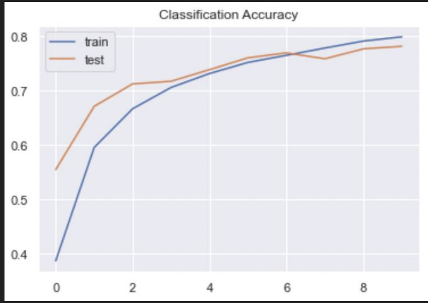
gaussian_filter

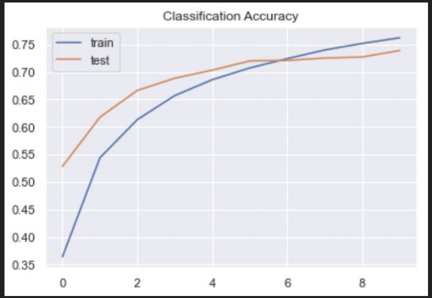
cv2.blur

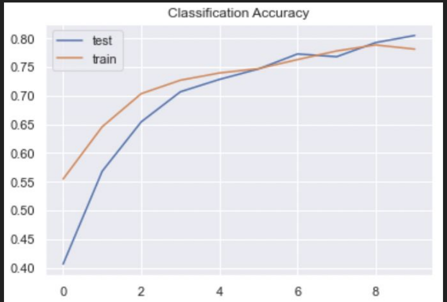


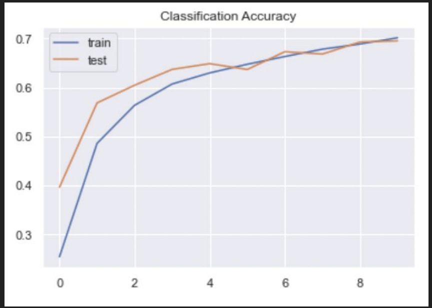


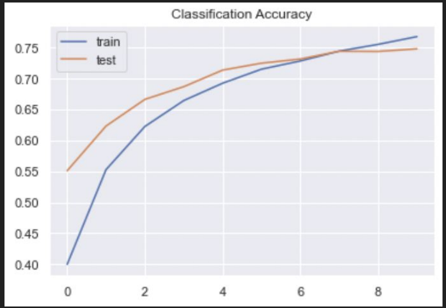
	Original	High Pass	Low Pass
Grey	Accuracy: 0.7815 Loss: 0.6574	Accuracy: 0.6684 Loss: 0.9668	Accuracy: 0.7389 Loss: 0.7620
RGB	Accuracy: 0.7782 Loss: 0.6643	Accuracy: 0.6949 Loss: 0.9119	Accuracy: 0.7482 Loss: 0.7404





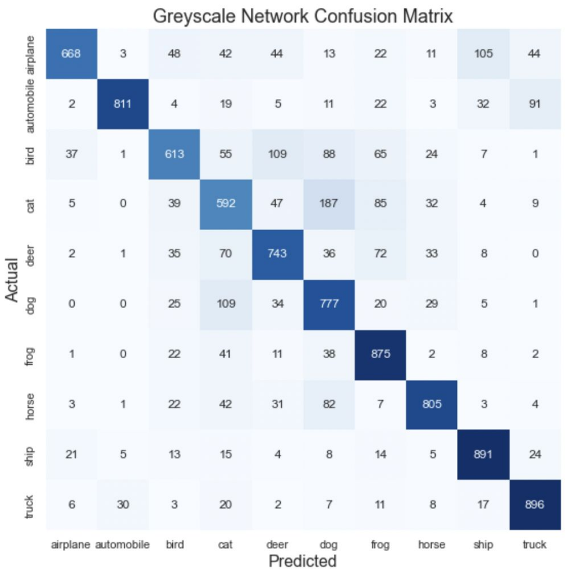




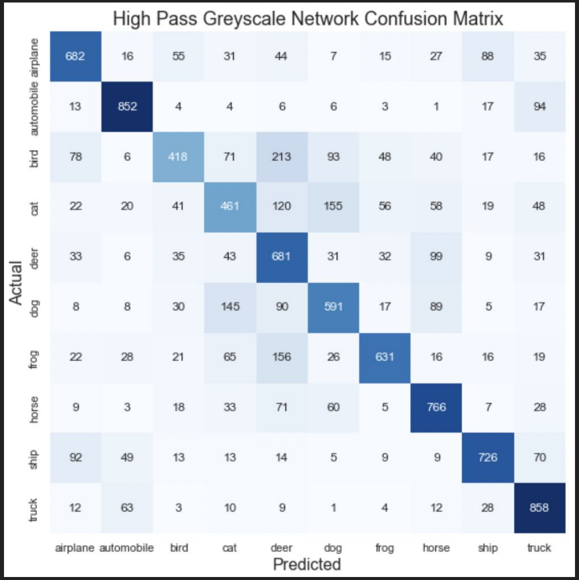


GREYSCALE

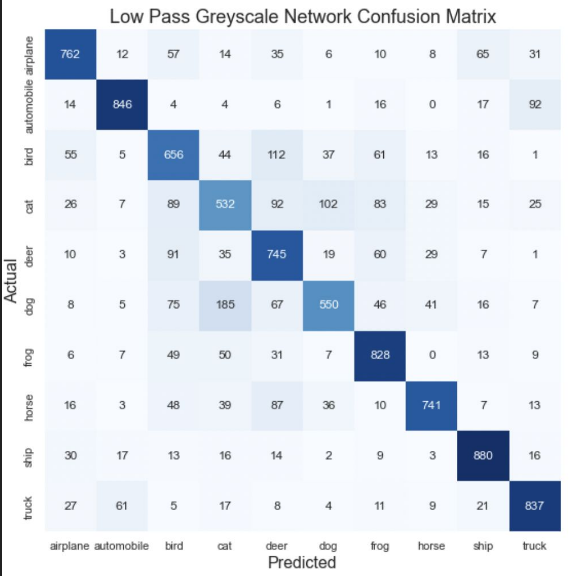
Original



High Pass

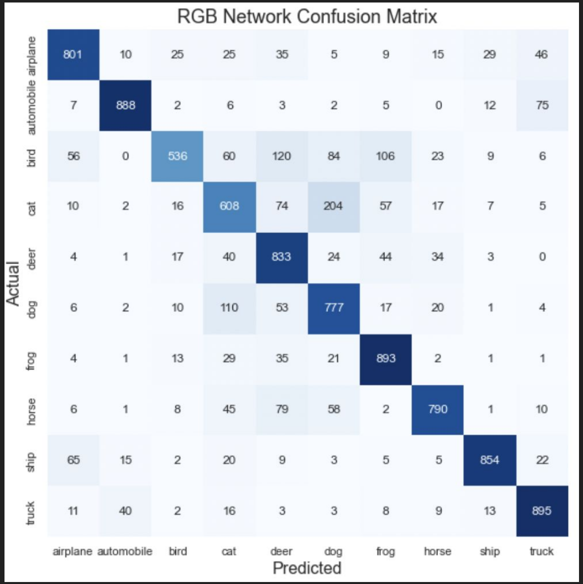


Low Pass

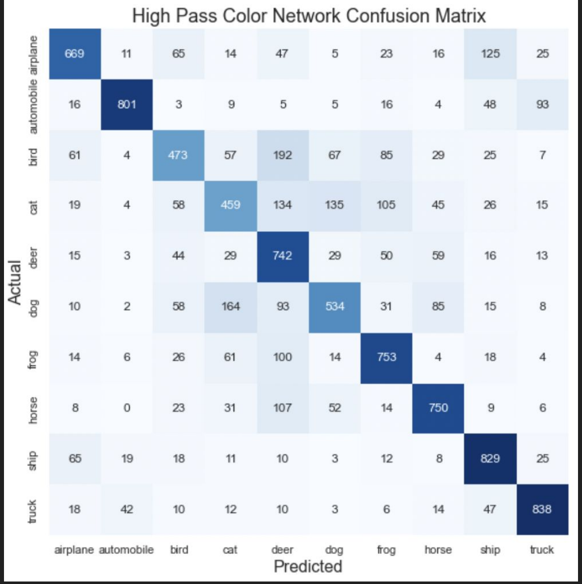


RGB

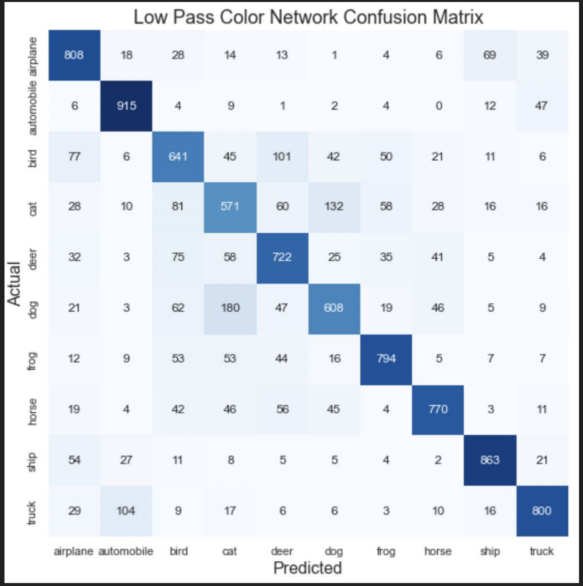
Original



High Pass



Low Pass



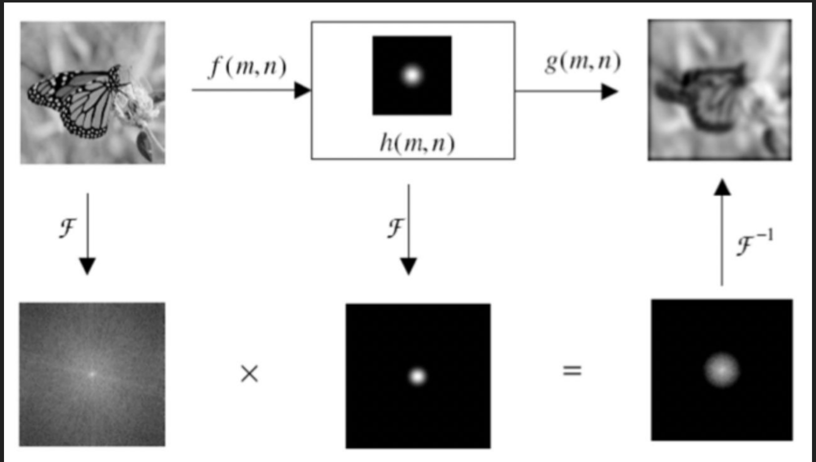
FFT

Image Processing in the Frequency Domain

The **Fourier Transform** is used to decompose an image into its sine and cosine components.

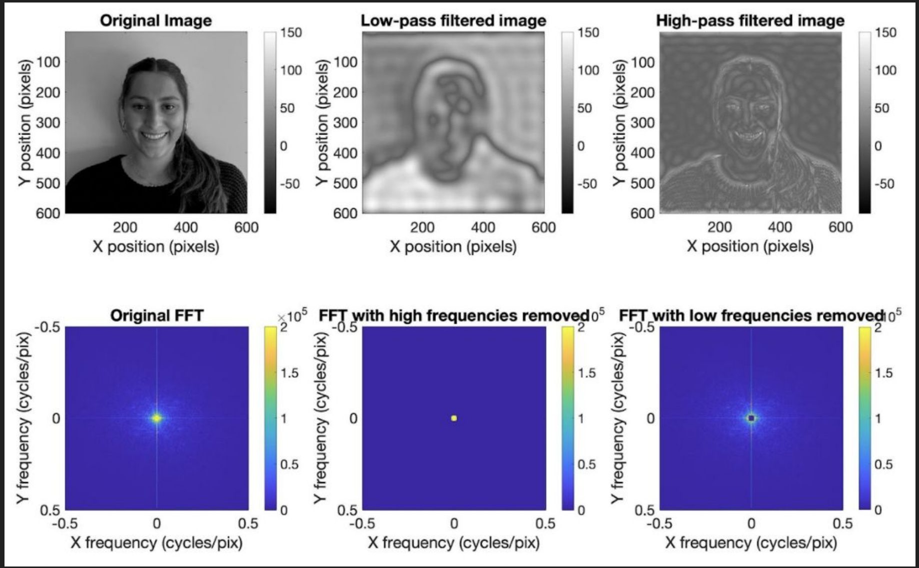
Theory

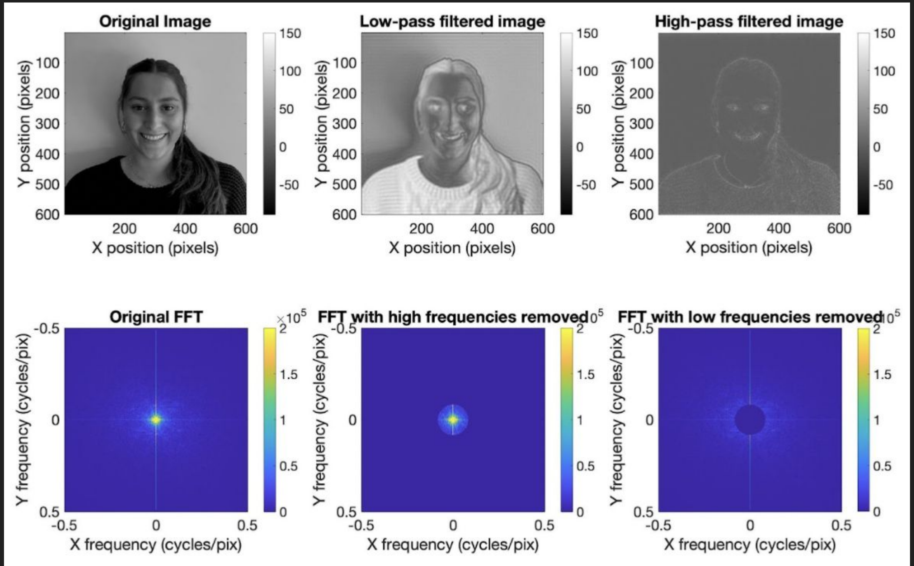
- A **Fast Fourier Transform** (FFT) is used for calculation of **2D Discrete Fourier Transform** (DFT) which is used to find the frequency domain.
- When working with image data, the edge points represent the high frequencies, since the amplitude of a signal varies drastically at that point.



Mask size

Changing mask size allows for regulation of the filtered frequencies, basically controls the range of frequencies that are removed. All values either inside or outside the desired radius are zeroed.





Why is it useful?

- Frequency domain gives you control over the whole images, where you can enhance (eg edges) and suppress (eg smooth shadow) different characteristics of the image very easily.

- Frequency domain has a established suit of processes and tools that be borrowed directly from signal processing in other domains.

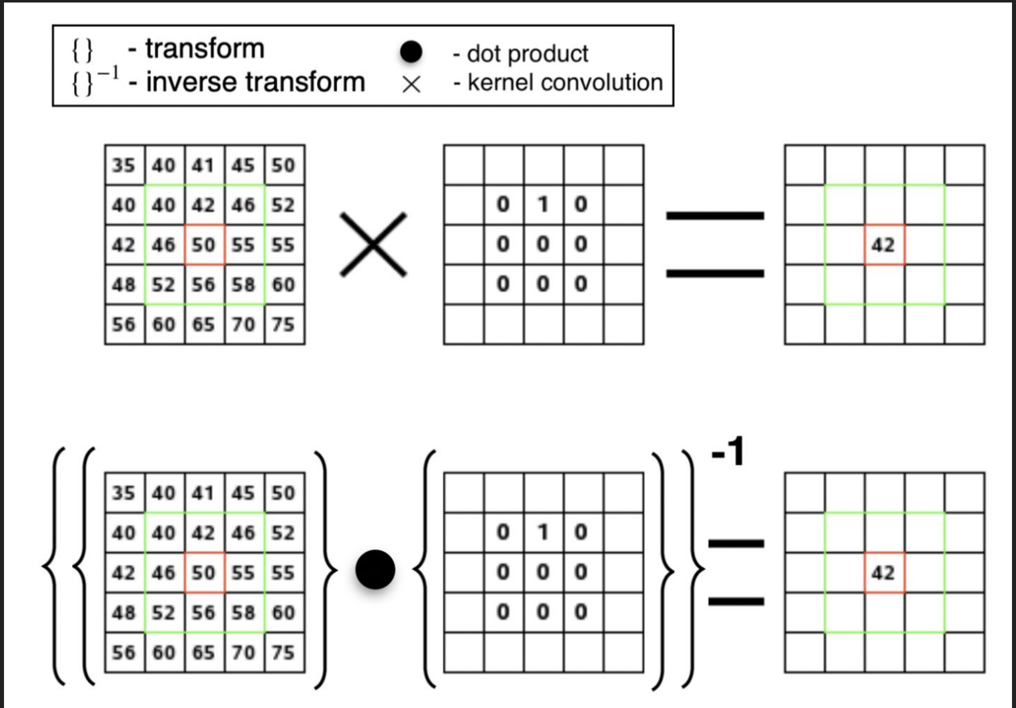
- Some tools used for even image recognition such as correlation, convolution etc are much simpler and computationally cheaper in frequency domain.

More FT fun facts

The dot product of x and y within the Fourier domain is the convolution of the Fourier domains of x and y.

In 2D, the Fourier operation costs $O(N^2 \log N^2) = O(N^2 \log N)$. Usually, the filter $w \in \mathbb{R}$^$K\times K$ is a *sliding window* over the image, for a total of N^2 (K,K) convolutions that cost $O(N^2 K^2)$. However, by padding w to be size $N\times N$ and computing, we get the following:

Researchers have reported significant reductions in training time **(<u>55s→ 0.4s</u>!!!)** with an improved $O(N^2K^2) \rightarrow O(N^2 \log N)$ convolution step.



$$x * y = F_N(F_N^{-1}(x) \bullet F_N^{-1}(y))$$