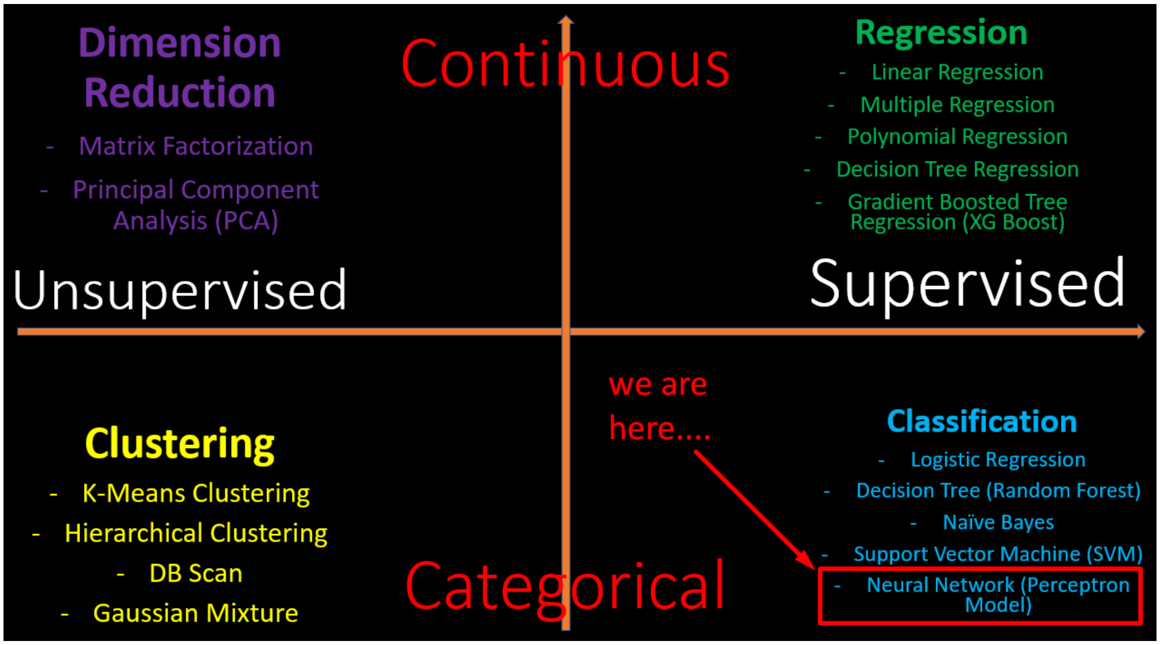
CONVOLUTIONAL NEURAL NETWORKS (CNN)

HOW IT WORKS
DR. ALVIN ANG

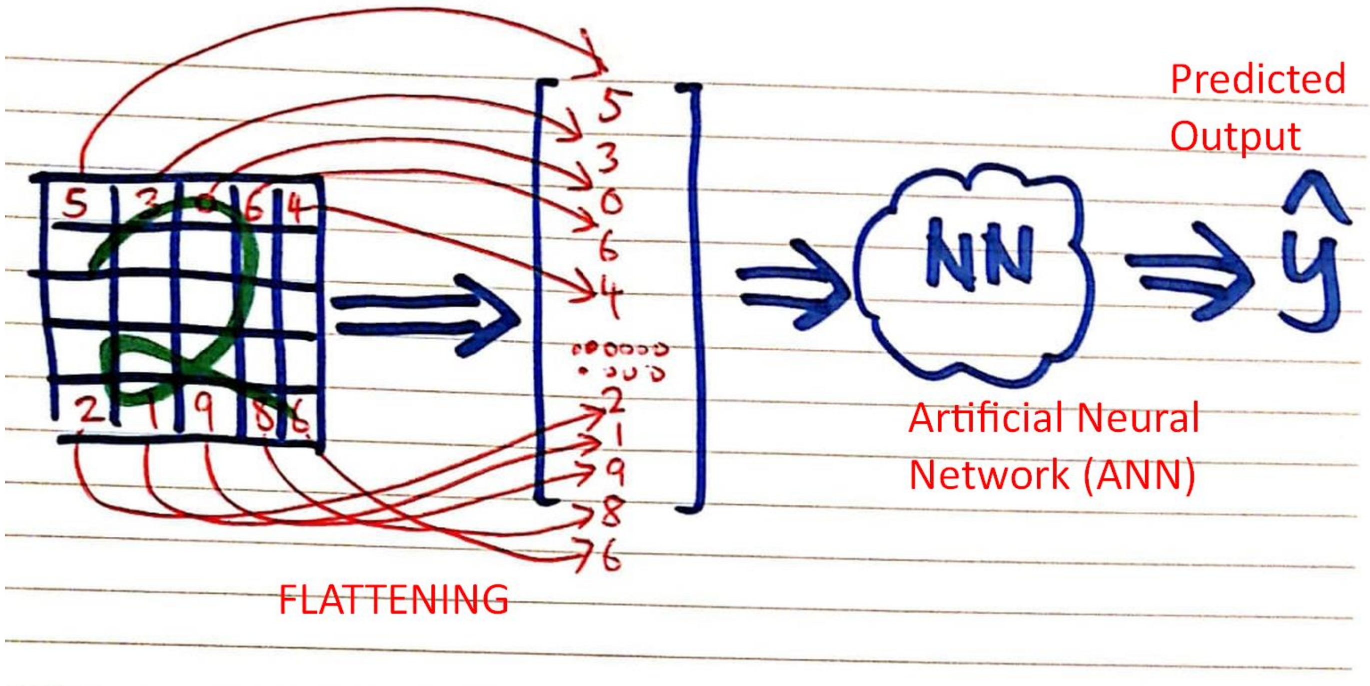
CONTENTS



- Above is a table categorizing the different Machine Learning algorithms.
- Objective of Neural Network is to predict a CATEGORY.

II. WHY CNN?

A. BRIEF RECAP OF ARTIFICIAL NEURAL NETWORK (ANN)

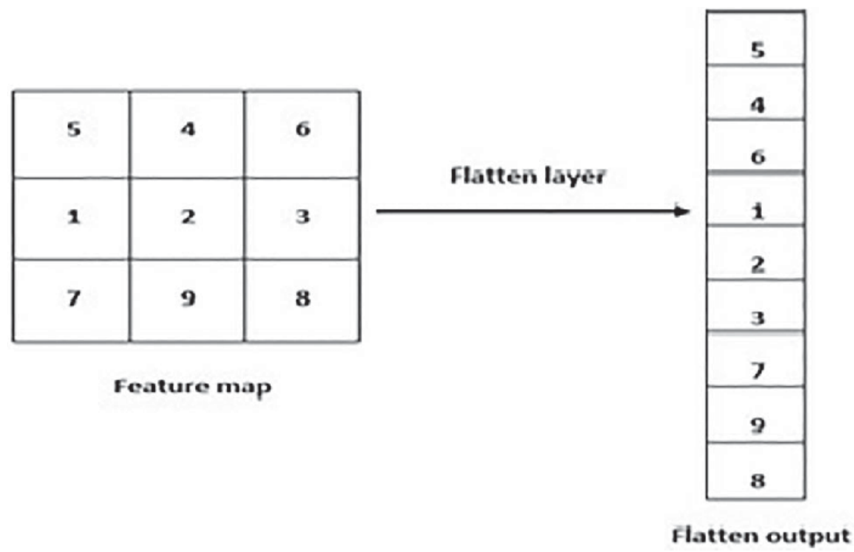


- In my previous article, I talked about how ANN worked (with images).
- An image "2" needs to be first FLATTENED, then fed into the NN for training / learning.
- Subsequently, once the NN is trained, it can predict new jpgs and output the label / classification.

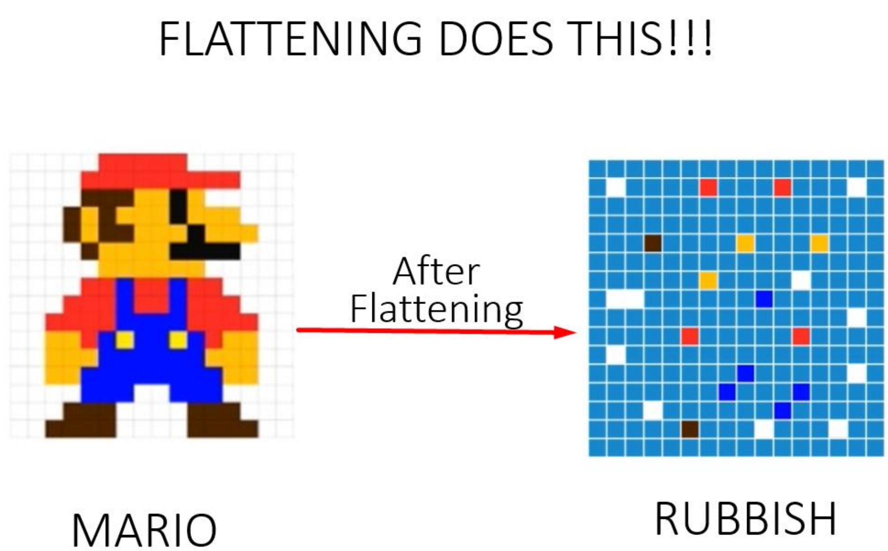
B. FLATTENING IS VERY BAD

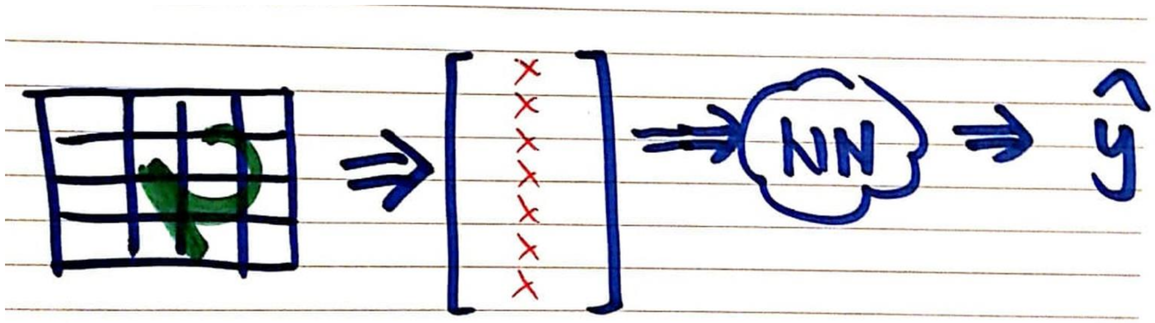
- However, FLATTENING is a very DESTRUCTIVE process.

- It breaks down the jpg into an array (which obviously you can't see a 2 anymore in the vector).



- Flattening converts a 3 x 3 array → 9 x 1 array



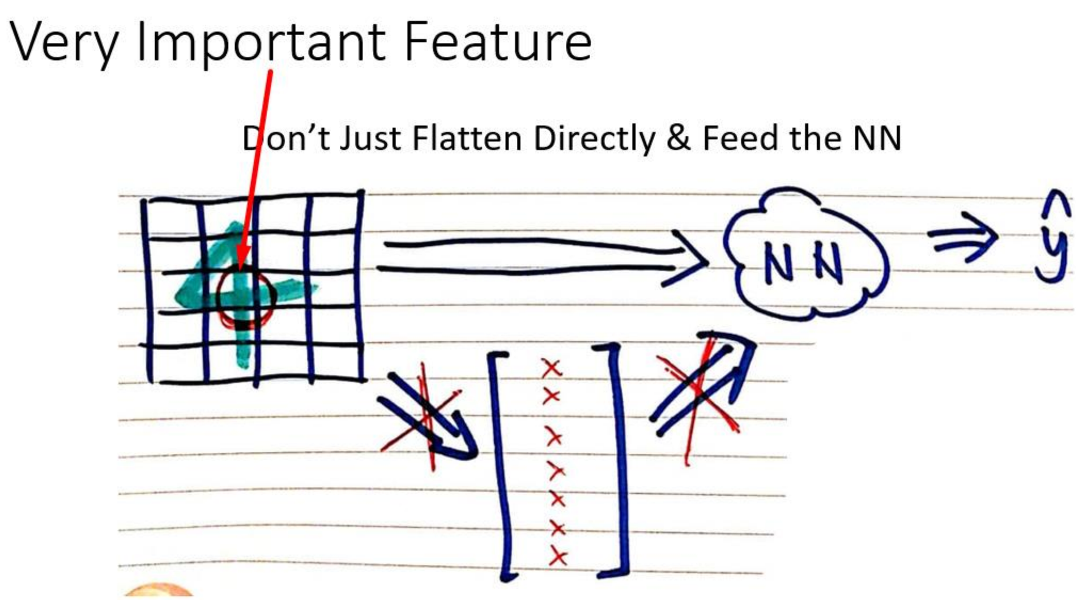


- Now if I write a 2 (tilted) like the above, the FLATTENED array obviously will be different from the first "2"'s array (the straight up one in the previous page).

- However, we all know a "2" is still just a "2" and they are both the same! But their arrays are completely different from one another!

- Thus, FLATTENING KILLS A LOT OF IMPORTANT INFORMATION.

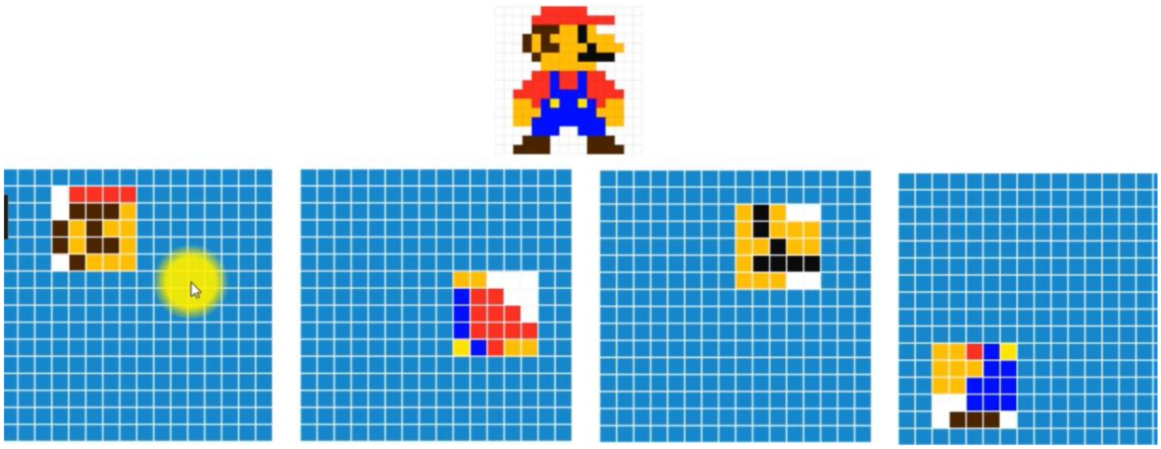
- We need a way to PREVENT INFORMATION LOSS.



- For example, a "4" important feature is the cross → without it, its hard for us to see it's a "4".

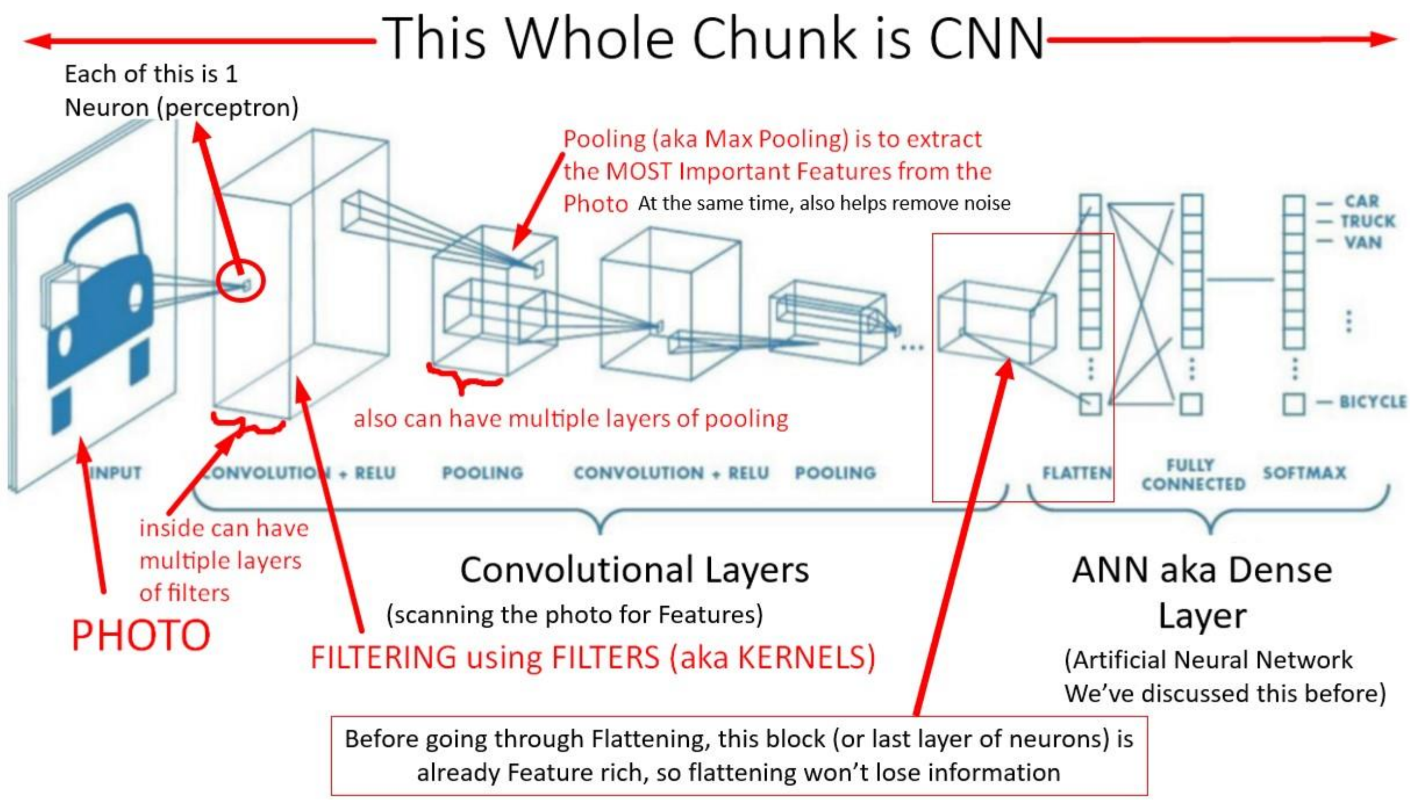
- We need a way to feed the IMPORTANT FEATURES into the NN for it to recognize the number.

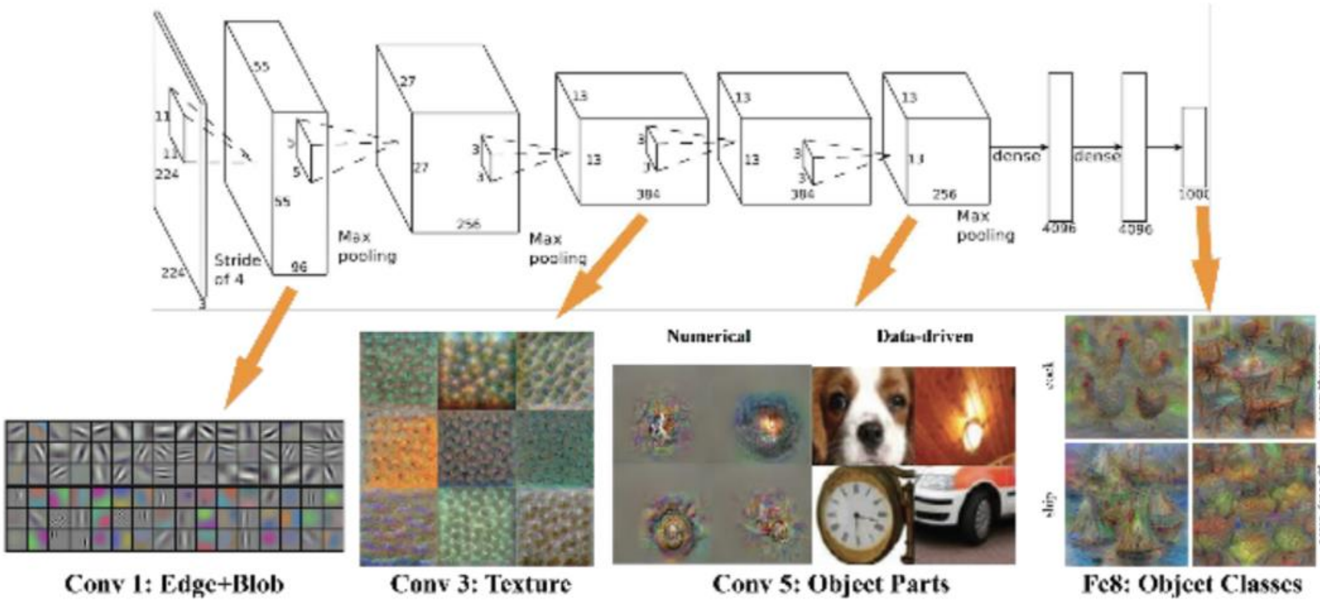
C. BASIC IDEA BEHIND CNN



- We need a way for the NN to pick up important features from the image (in order to recognize it).

- Mario's Ear… + Moustache… + Shoulder… each part will piece together to help the NN recognize that its Mario.

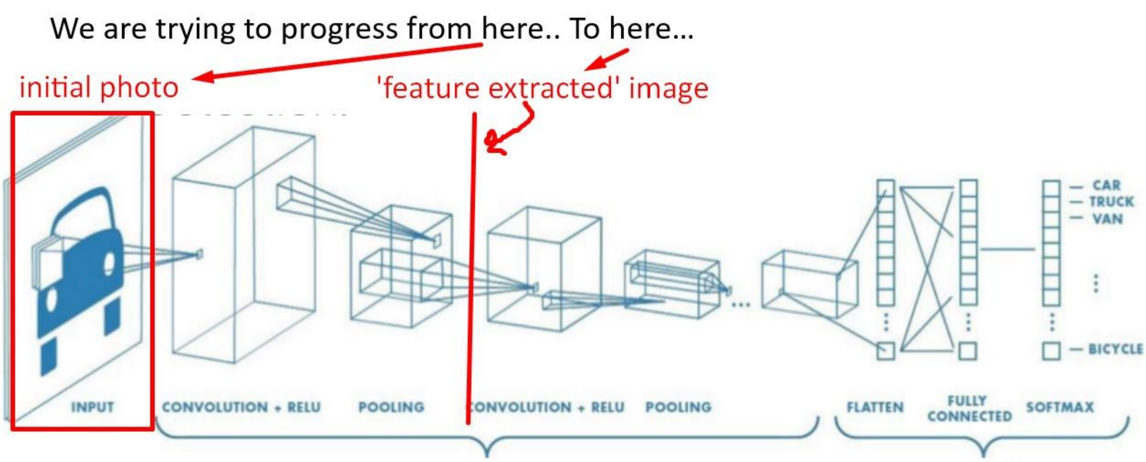

This Whole Chunk is CNN
Each of this is 1
Neuron (perceptron)
Pooling (aka Max Pooling) is to extract
the MOST Important Features from the
Photo At the same time, also helps remove noise
CAR
TRUCK
VAN
BICYCLE
also can have multiple layers of pooling
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
inside can have
multiple layers
of filters
PHOTO
Convolutional Layers
(scanning the photo for Features)
FILTERING using FILTERS (aka KERNELS)
ANN aka Dense
Layer
(Artificial Neural Network
We've discussed this before)
Before going through Flattening, this block (or last layer of neurons) is
already Feature rich, so flattening won't lose information



- At every Convolution layer, Features are extracted.
- 1st Convolution normally only extracts very simple Features like Vertical and Horizontal Lines.
- Subsequent layers search for more complex Features like Texture and Parts of an Object.

IPYNB:

https://www.alvinang.sg/s/Understanding_CNN_by_Dr_Alvin_Ang.ipynb



A. STEP 1: IMPORT ALL LIBRARIES

Step 1: Import All Libraries

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D

import numpy as np
import matplotlib.pyplot as plt
import cv2  # only used for loading the image

%matplotlib inline
```

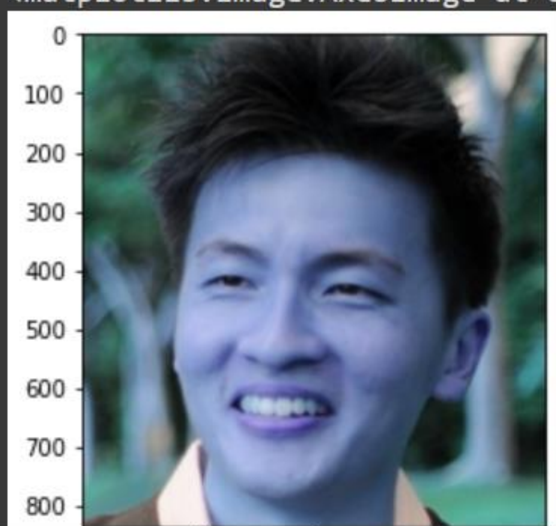
B. STEP 2: LOAD THE PHOTO

Step 2: Show the Photo

```
[ ] alvin = cv2.imread('/content/ALVIN.jpg')
```

```
plt.imshow(alvin)
```

```
<matplotlib.image.AxesImage at 0x7f23e250e8d0>
```



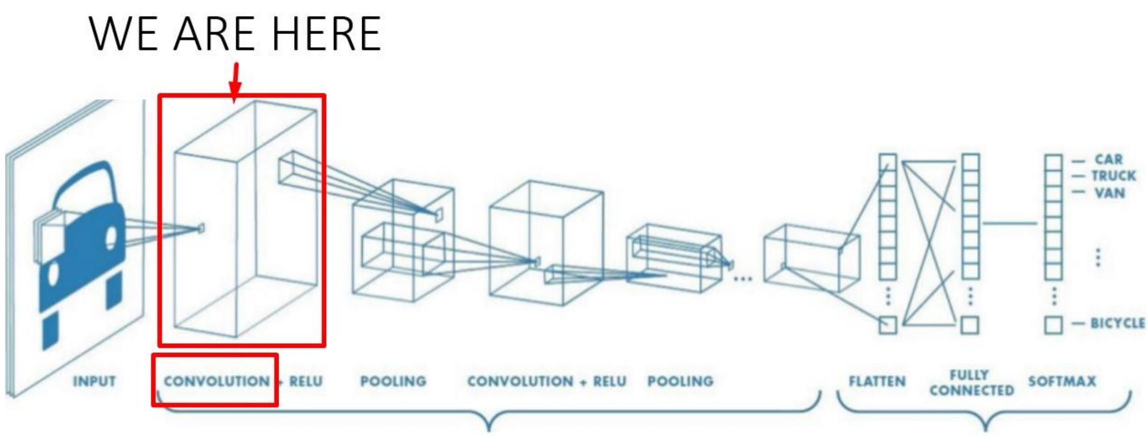

```
# what does the image look like?
alvin.shape

#1408 pixels by 793 pixels with 3 channels RGB
```

```
(836, 789, 3)
```

C. STEP 3: WHAT IS A CONVOLUTION?

Step 3: Demonstrating effect of 1 Convo Layer

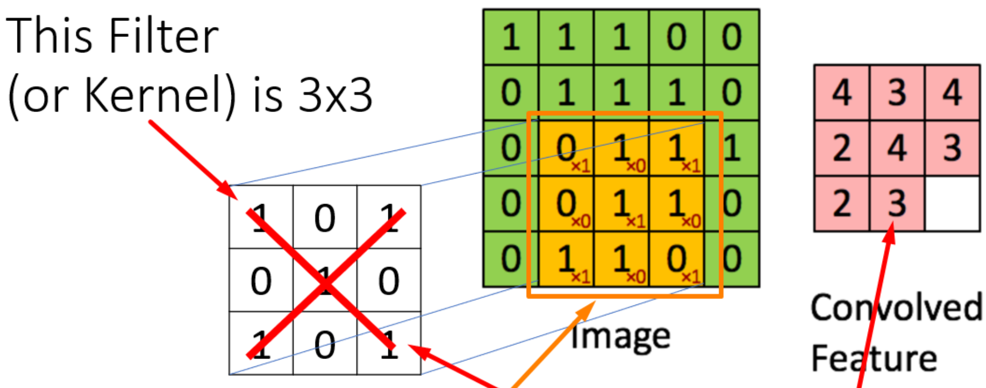
What Does a Convolution Layer Do? aka Filtering



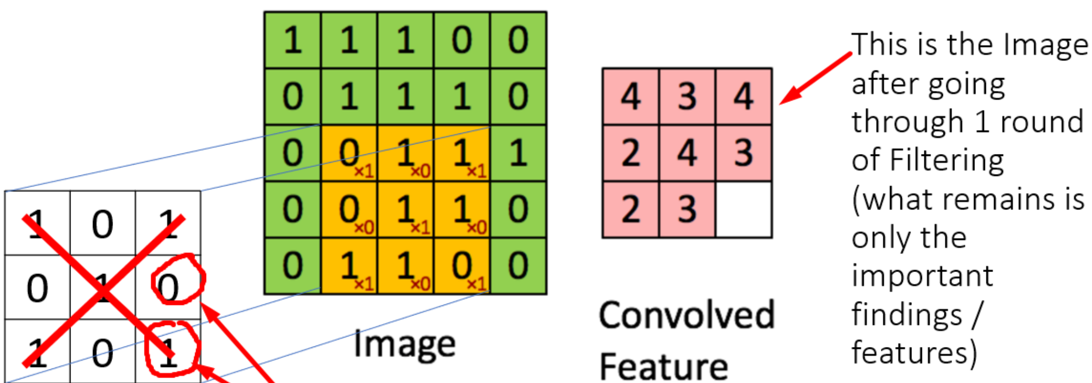
1. CONVOLUTION DOES THIS

To see animated .gif, go here to understand how the Filter slides across the image:

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53



- Filter (or Kernel) is trying to seek out an 'X' Feature
- Imagine that particular yellow portion of the image is all 1
- Then (1x1) + (1x1) + (1x1) +…. Will get you MAXIMUM number 9 → means that 'X' feature is found!
- Imagine that is all 0… then (0x1) + (0x1) +…. You will get 0 → No 'X' feature there!



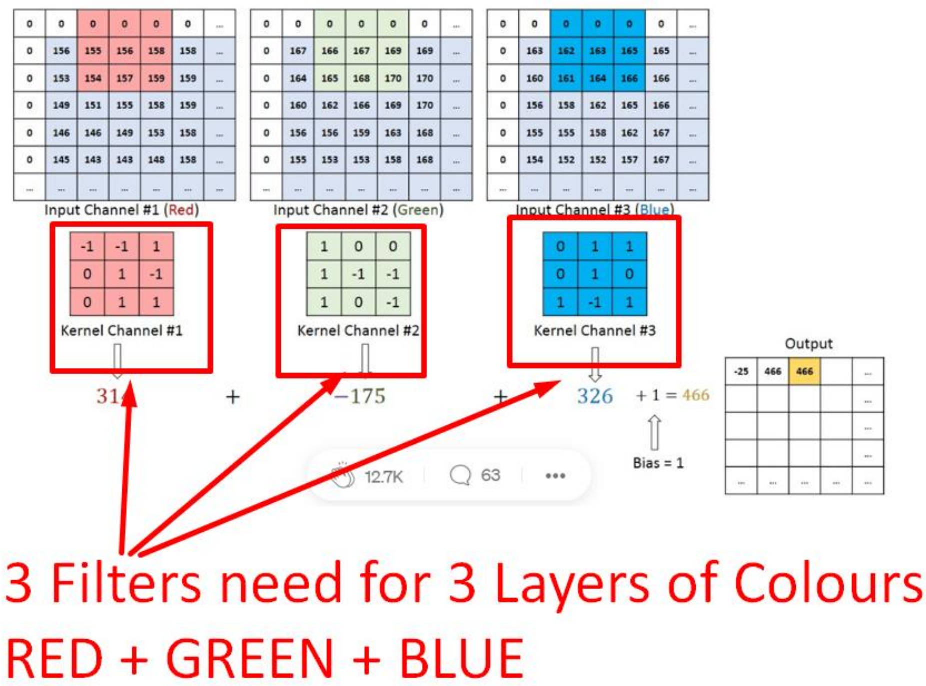
- Note that these are Weights which will be automatically found by the CNN (updated with each Epoch)
- In other words, CNN will find out the "best" features to look out for automatically (in each Filter).

2. SEQUENTIAL MODELING

```
model = Sequential()

model.add(Conv2D(filters = 3,
                 #we need to use 3 filters because its RGB

                 kernel_size = (3, 3),
                 #we use a commonly used kernel_size of 3x3

                 #strides = 1,
                 #default stride = 1, even if you don't declare

                 #padding = 'same',
                 #default padding = 'valid', even if you don't declare
                 #we use 'same' to prevent loss of information

                 #activation = None,
                 #default activation = 'none', even if you don't declare
                 #but for now, we purposely force our activation to 'none'
                 #for purpose of demonstrating the effect of 1 Conv layer
                 #without activation

                 input_shape = alvin.shape))
```

```
model = Sequential()

model.add(Conv2D(filters = 3,
                 #we need to use 3 filters because its RGB

                 kernel_size = (3, 3),
                 #we use a commonly used kernel_size of 3x3
```

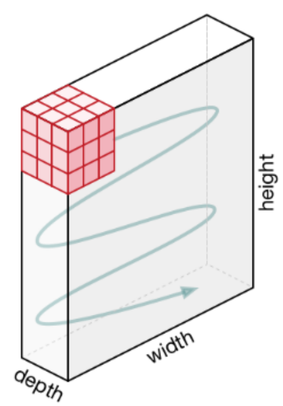
- We need 3 filters because our ALVIN.jpg is made up of 3 primary colours = RED + GREEN + BLUE



To see animated .gif, go here to understand how the 3 Filters slide across the image:

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

So basically, this happens…..



Movement of the Kernel

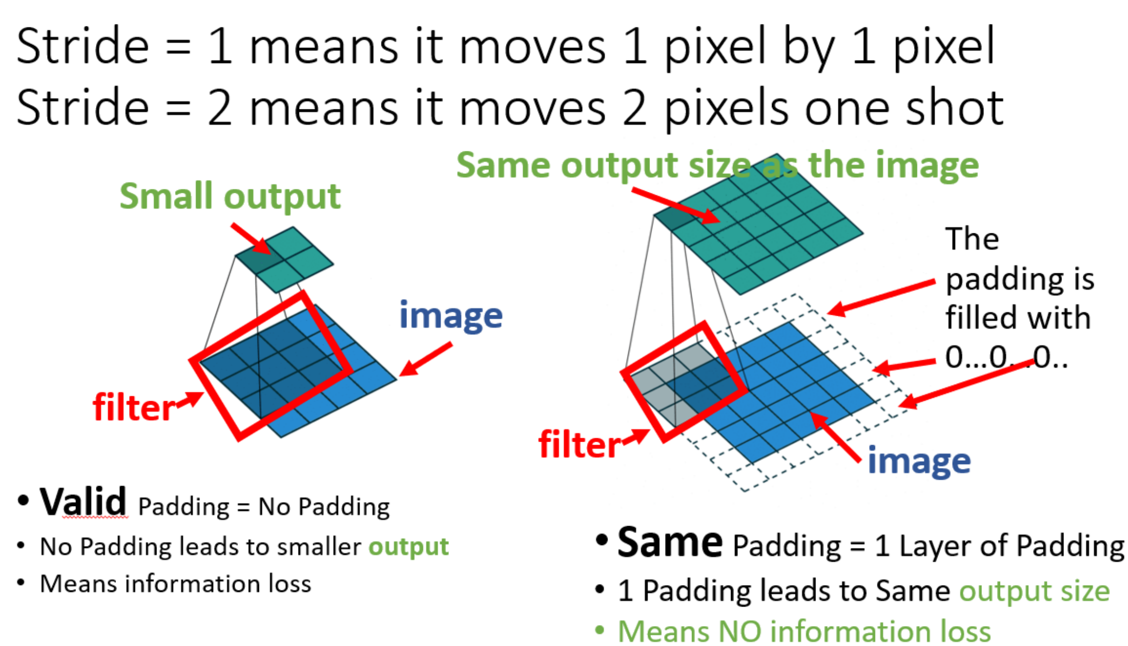
3. WHAT IS PADDING? WHY DO WE NEED IT?

```
model.add(Conv2D(filters = 3,
                 #we need to use 3 filters because its RGB

                 kernel_size = (3, 3),
                 #we use a commonly used kernel_size of 3x3

                 #strides = 1,
                 #default stride = 1, even if you don't declare

                 #padding = 'same',
                 #default padding = 'valid', even if you don't declare
                 #we use 'same' to prevent loss of information
```

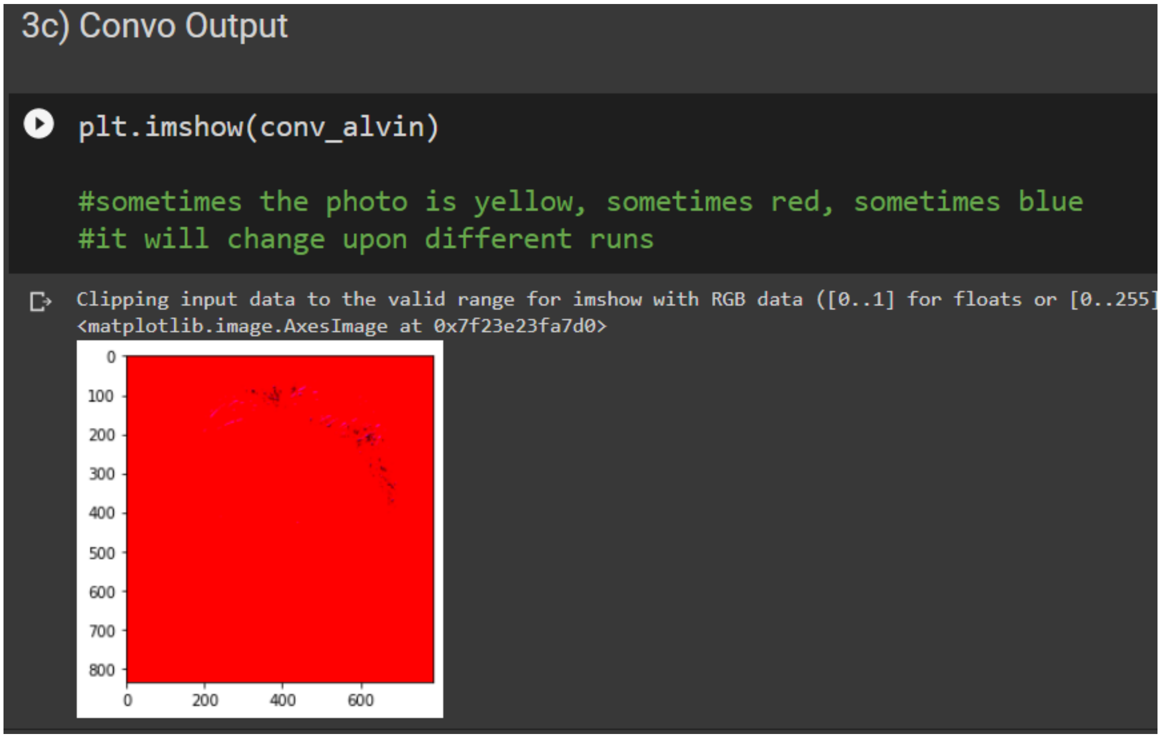
To see animated .gif, go here to understand how padding is done:

https://github.com/vdumoulin/conv_arithmetic

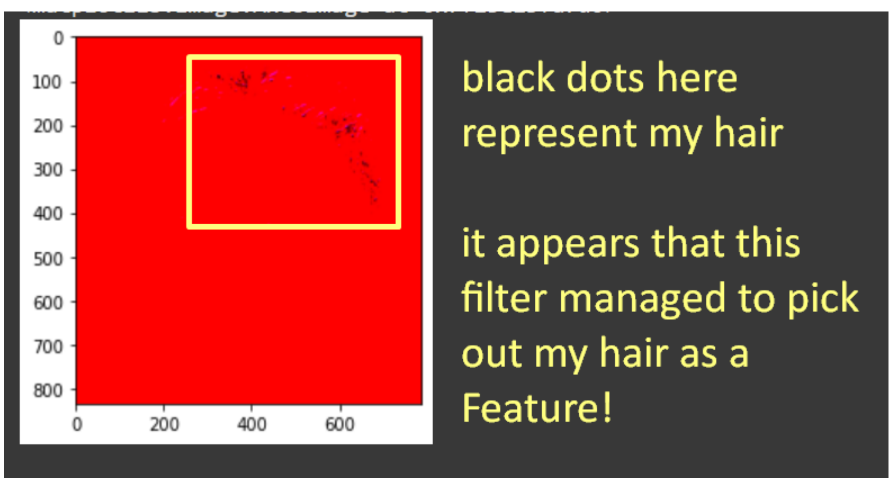
https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53



4. CONVOLUTION OUTPUT / EFFECT OF CONVOLUTION



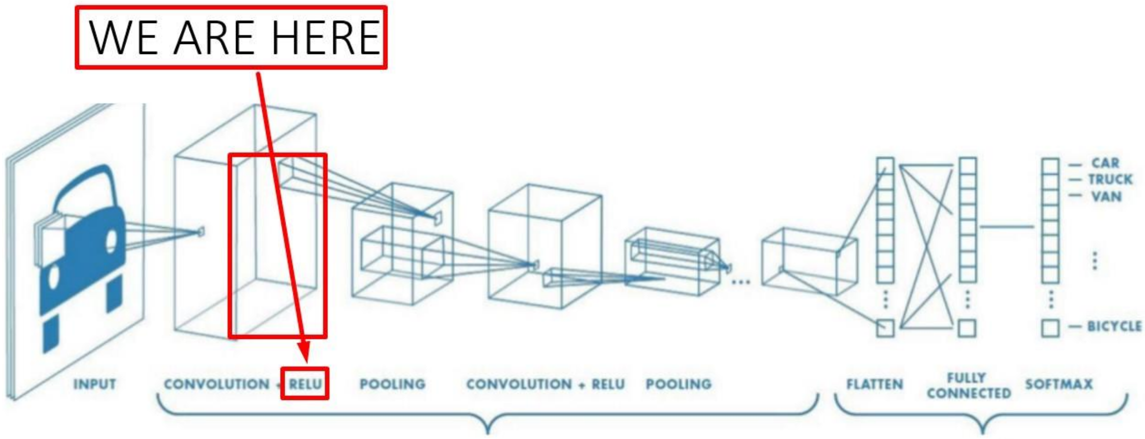
- What do we see there?
- After 1 x Convolution with 3 filters, the entire image turned RED! (but you can try running it several times… it will change colours)



D. STEP 4: WHAT IS THE EFFECT OF USING RELU ACTIVATION?

Step 4: Demonstrating effect of adding Relu activation

What happens when you Add a Relu Activation?



- Relu is an activation function and has already been explained here: https://www.alvinang.sg/s/Artificial-Neural-Network-ANN-How-It-Works-by-Dr-Alvin-Ang.pdf

1. SEQUENTIAL MODELING

4a) Sequential Modeling

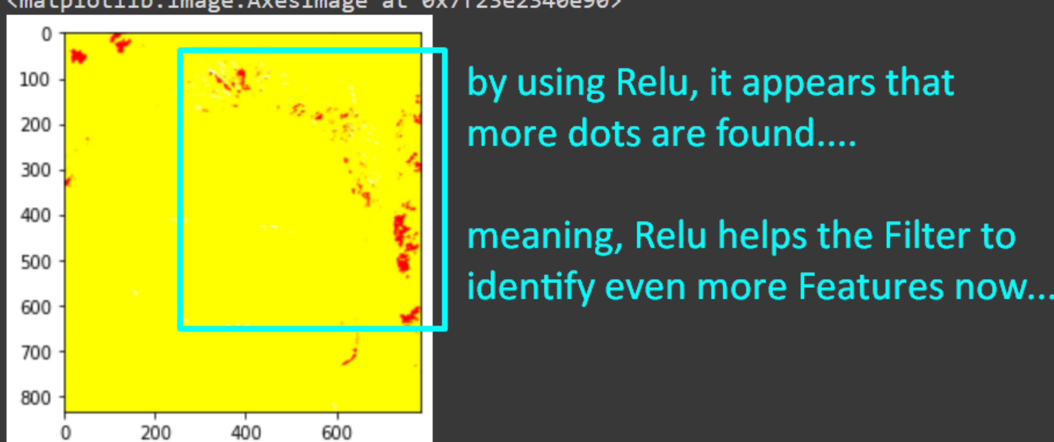
```
model2 = Sequential()

model2.add(Conv2D(filters = 3,
                  kernel_size = (3, 3),
                  #strides = 1,
                  #padding = 'same',

                  activation = 'relu',
                  #now we purposely change to Relu Activation
                  #to observe its effect

                  input_shape = alvin.shape))
```

2. RELU OUTPUT / EFFECT OF USING RELU ACTIVATION

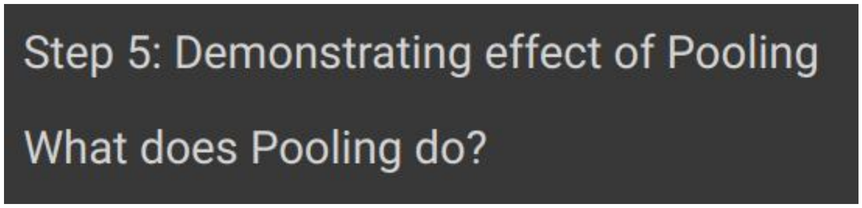
4c) Relu Output

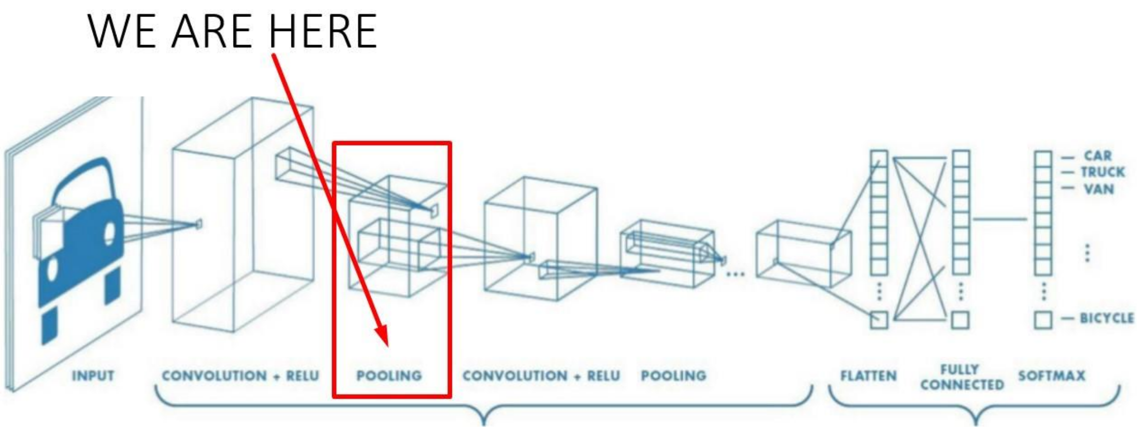
```
plt.imshow(conv_alvin_2)
```

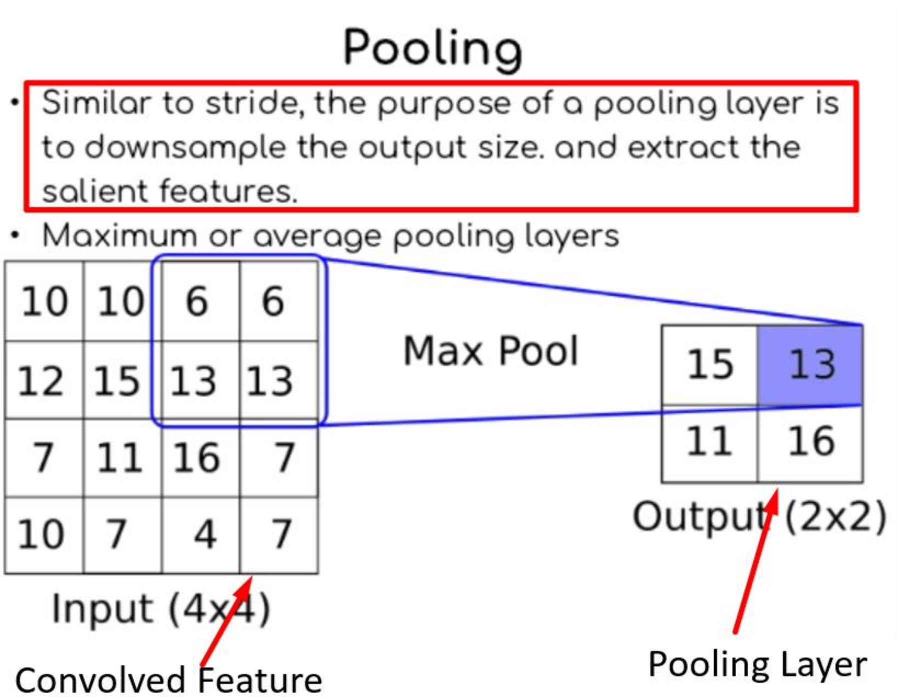
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floa
<matplotlib.image.AxesImage at 0x7f23e2340e90>
```



E. WHAT IS POOLING?

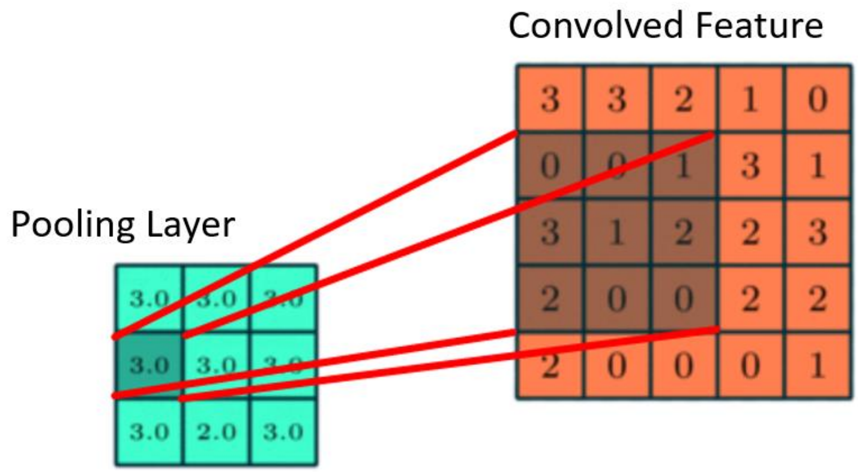


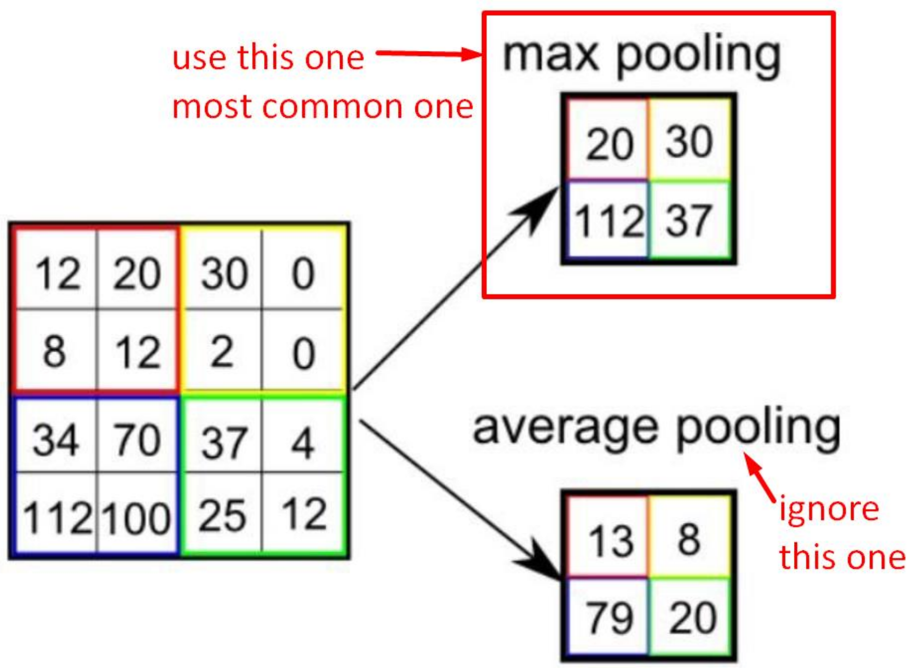




To see animated .gif, go here to understand how the Pooling Layer slides across the Convolved Feature:

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53





1. SEQUENTIAL MODELING

5a) Sequential Modeling

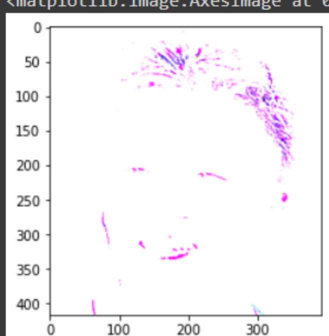
```
[ ] model3 = Sequential()

    model3.add(Conv2D(filters = 3,
                      kernel_size = (3, 3),
                      #strides = 1,
                      #padding = 'same',

                      activation = 'relu',
                      #activation is set to None by default
                      #but over here, we want to progressively see the effects
                      #of Convolution--> Relu --> Pooling
                      #so we use Relu

                      input_shape = alvin.shape))

    model3.add(MaxPooling2D((2, 2)))
    #2x2 receptive field is default and commonly used
```

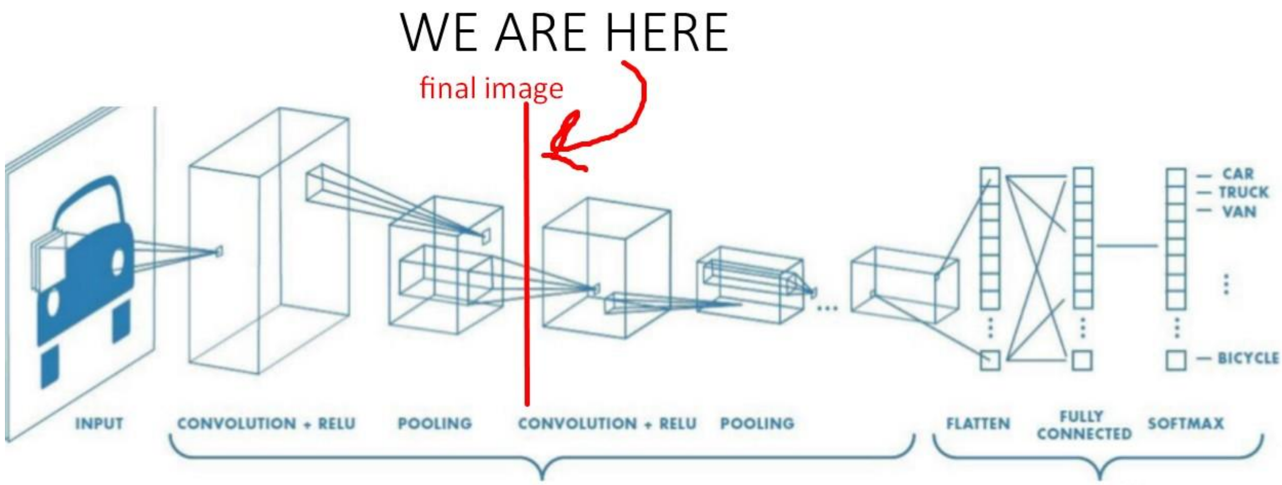
pooling layer is added

2. POOLING OUTPUT / EFFECT OF POOLING

```
plt.imshow(conv_alvin_3)

#you can see that the Convo + Relu + Pooling extracted
#my hair + eyes + teeth
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
<matplotlib.image.AxesImage at 0x7f23e23cd590>
```




WE ARE HERE
final image
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
CAR
TRUCK
VAN
BICYCLE

REFERENCES

- https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

ABOUT DR. ALVIN ANG

Dr. Alvin Ang earned his Ph.D., Masters and Bachelor degrees from NTU, Singapore. He is a scientist, entrepreneur, as well as a personal/business advisor. More about him at www.AlvinAng.sg.