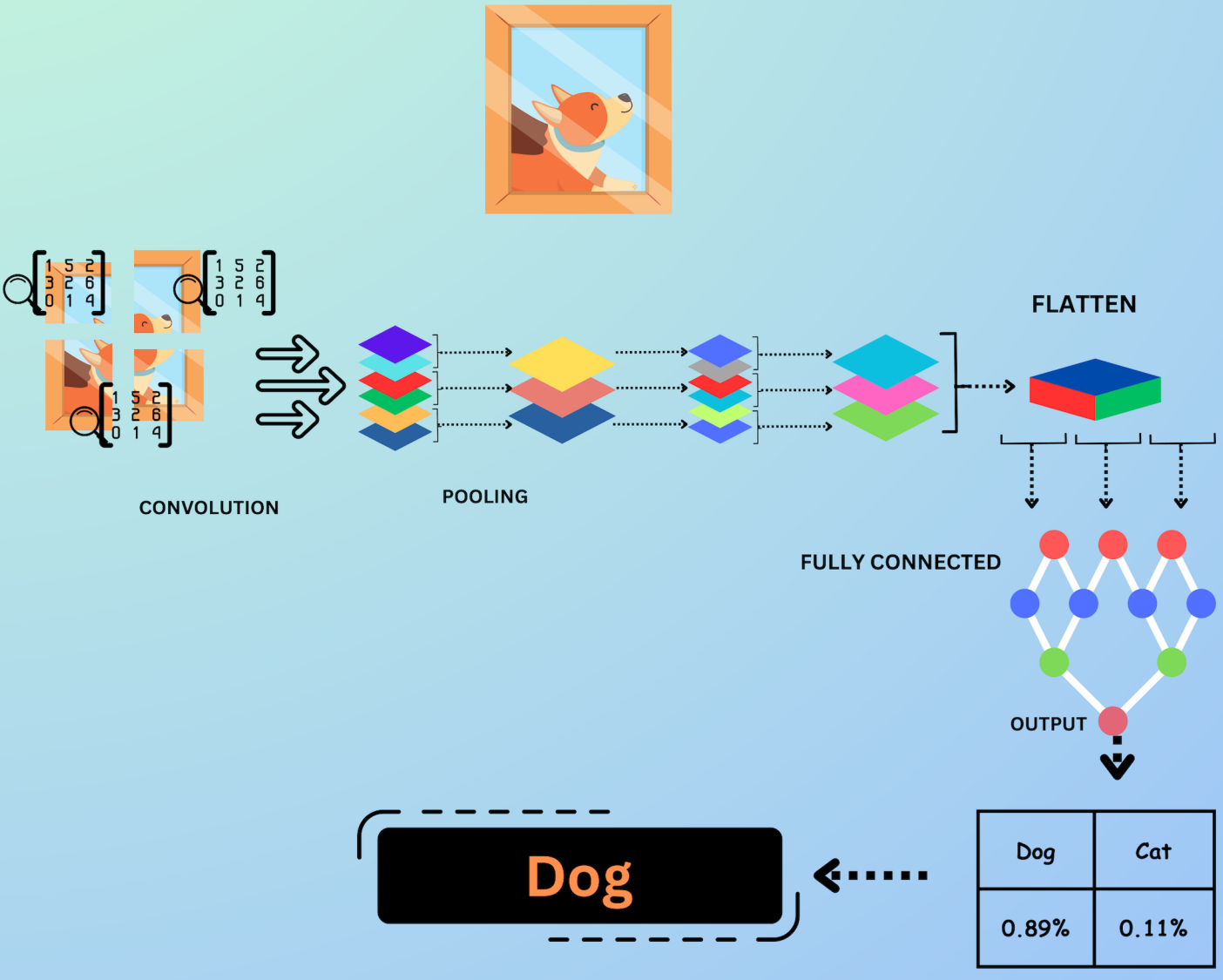
FUNDAMENTALS OF DEEP LEARNING

Structure of Convolutional Neural Networks



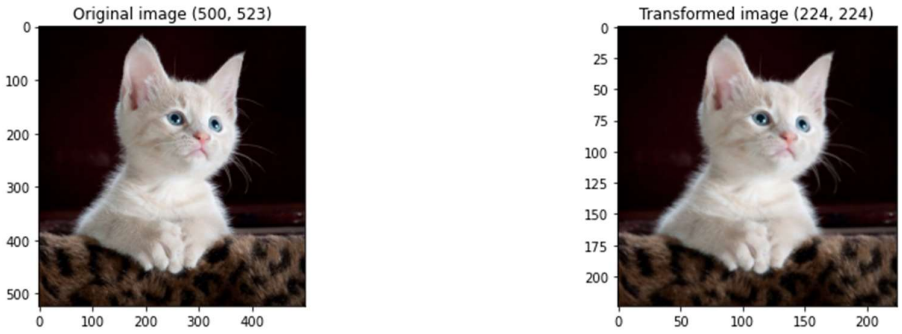
by Shivang Kainthola

Before we build a CNN…

Before we build a CNN, **the image data we have must be pre-processed** to ensure it is ready to be used by the neural network.

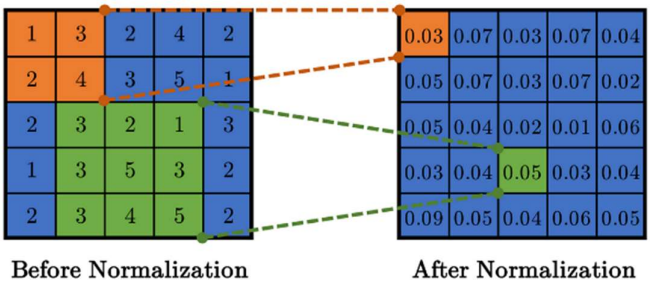
Pre-processing of image data for CNNs involves steps like:

1) Re-sizing images



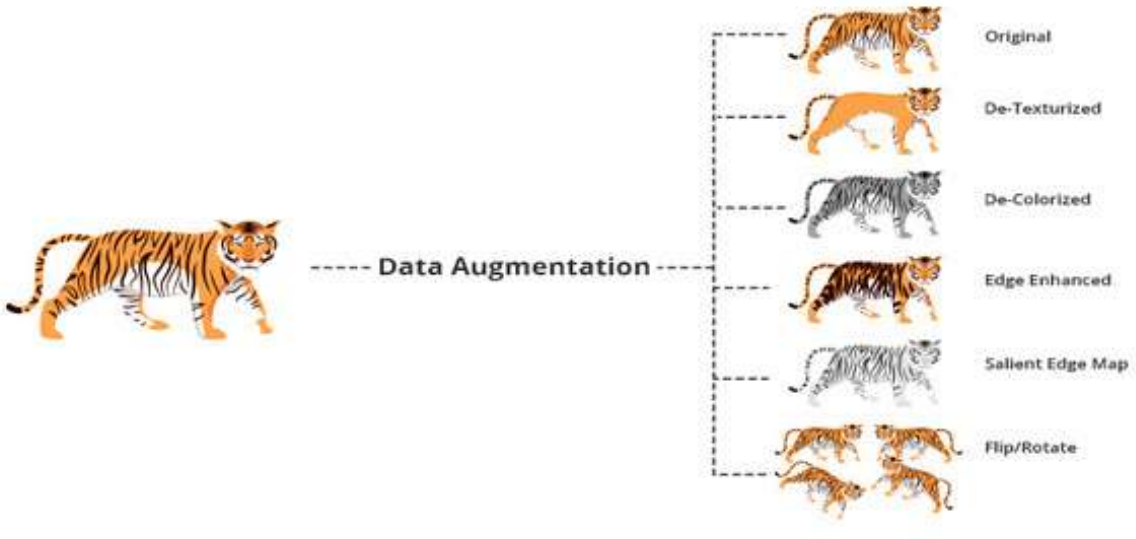
→ The images of different sizes are converted to one uniform size (usually to 224 x 224).

2) Normalizing the pixel data of images



→ Since pixels can have varying values of density, we normalize them to one range, usually between 0 and 1, or -1 and +1.

3) Data Augmentation :



→ Data augmentation is the process of generating more training examples (images) by rotating, flipping, lighting changes, and applying similar techniques on existing images.

⟶ The training data is thus increased from existing data, and the CNN will work on variations of the same image and generalize better.

Pre-processing of images can also be done within the neural network by using the *tf.keras.preprocessing.image* module,

with which we can define some layers which process the images before they reach the convolutional layer.

Structure of Convolutional Neural Networks

The structure of a convolutional neural network usually consists of the following layers :

1) **Convolutional Layers** : Apply kernels to extract features from the image and generate feature maps.

2) **Pooling Layers** : Extract information from feature maps and reduce their size.

3) **Flatten Layer** : To convert multi-dimensional feature maps into 1-D vectors.

4) **Fully Connected Layer** : To process the flattened image data and carry out the classification.

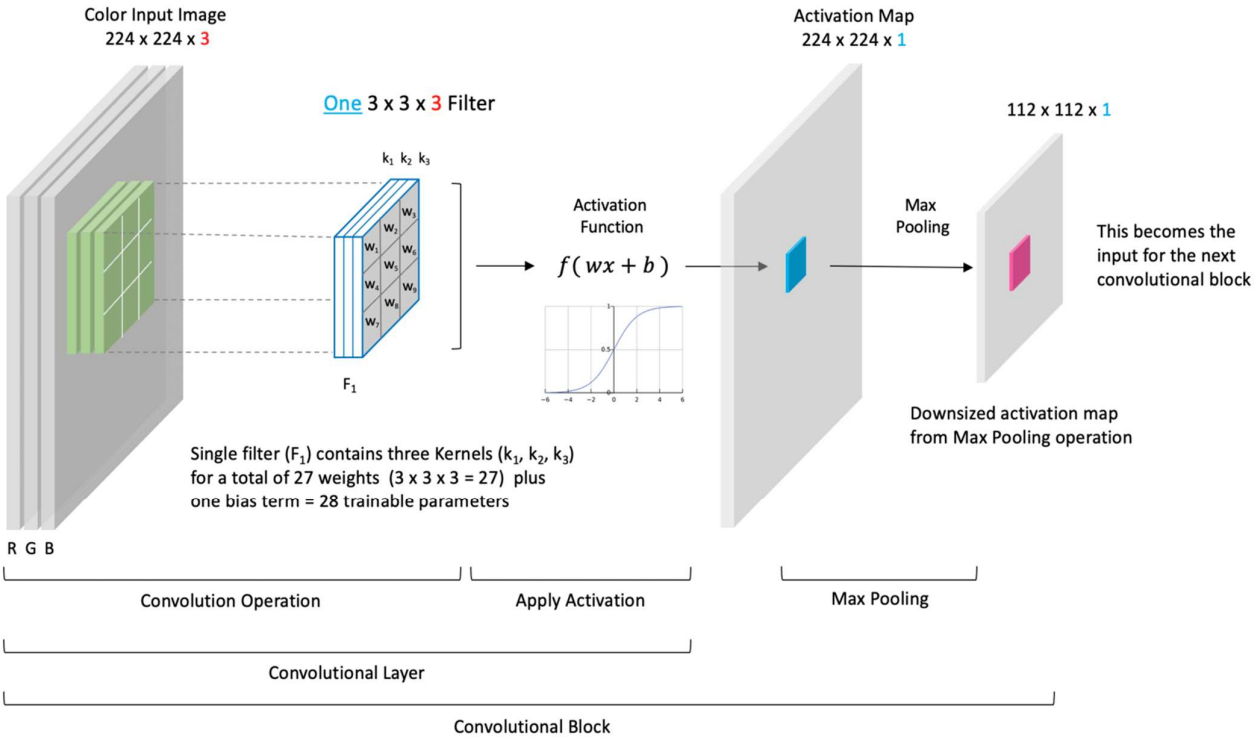
! **Batch Normalization Layer** : To normalize inputs between layers.

! **Dropout Layer** : Drop out a fraction of neurons from a layer.

Let us examine the role of each layer in detail :

1) Convolution Layer

⟶ This layer applies kernels to images, extracts features from them and generates feature maps.



https://learnopencv.com/understanding-convolutional-neural-networks-cnn/

⟶ It includes a layer with activation functions for the convolutional layer.

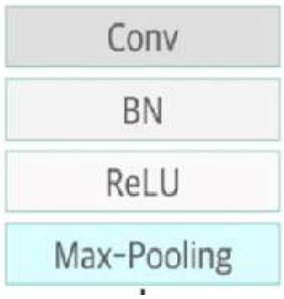
⟶ A convolution layer combined with batch normalization, pooling, dropout layers is called a convolutional block.

⟶ Depending on the task, multiple convolutional blocks are required and placed.

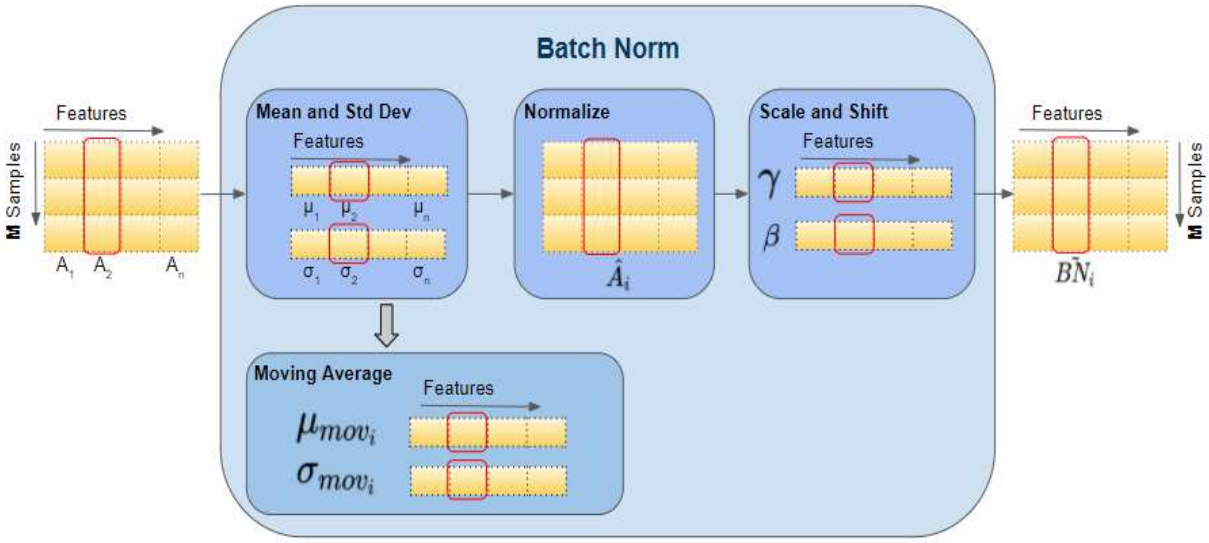
2) Batch Normalization Layer :

⟶ It normalizes the inputs between layers, can be placed before or after the activation function layer.

! It is optional, and used in very deep neural networks, as a regularization technique.



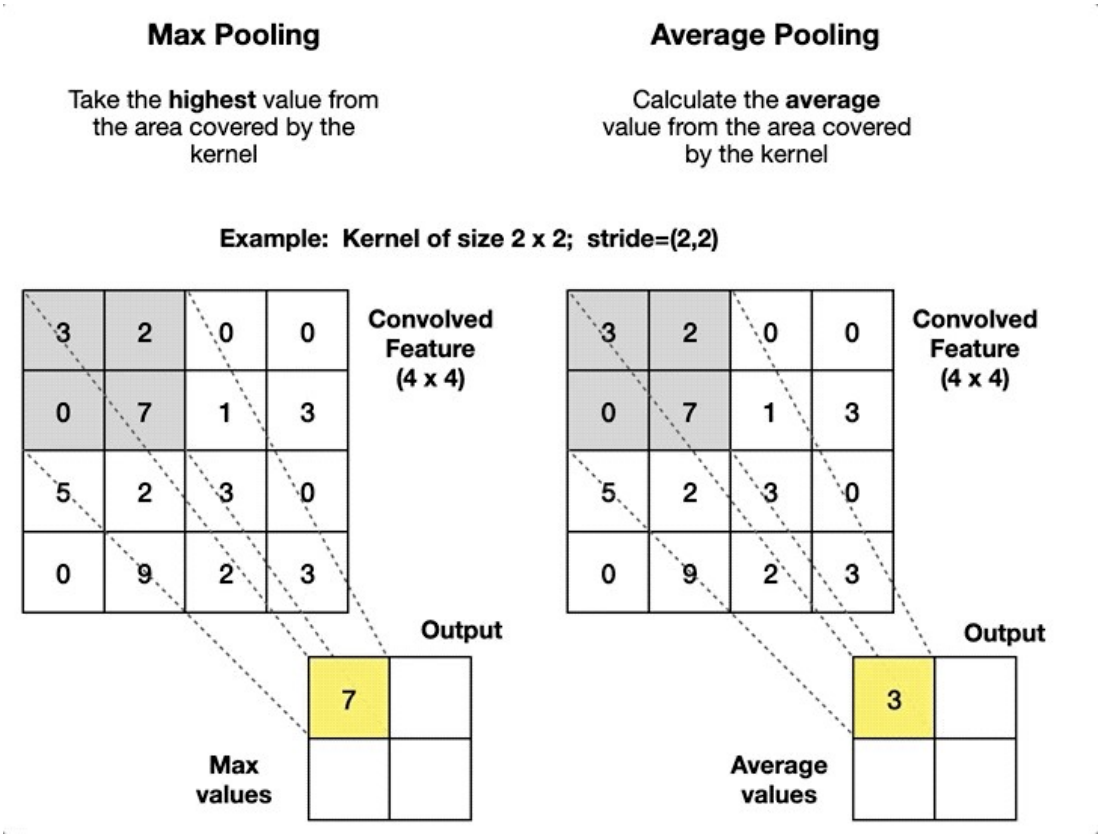
https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739



3) Pooling Layer :

⟶ Extract information from feature maps and shrinks their size.

⟶ This layer can apply different types of pooling methods, like average pooling, max pooling etc.



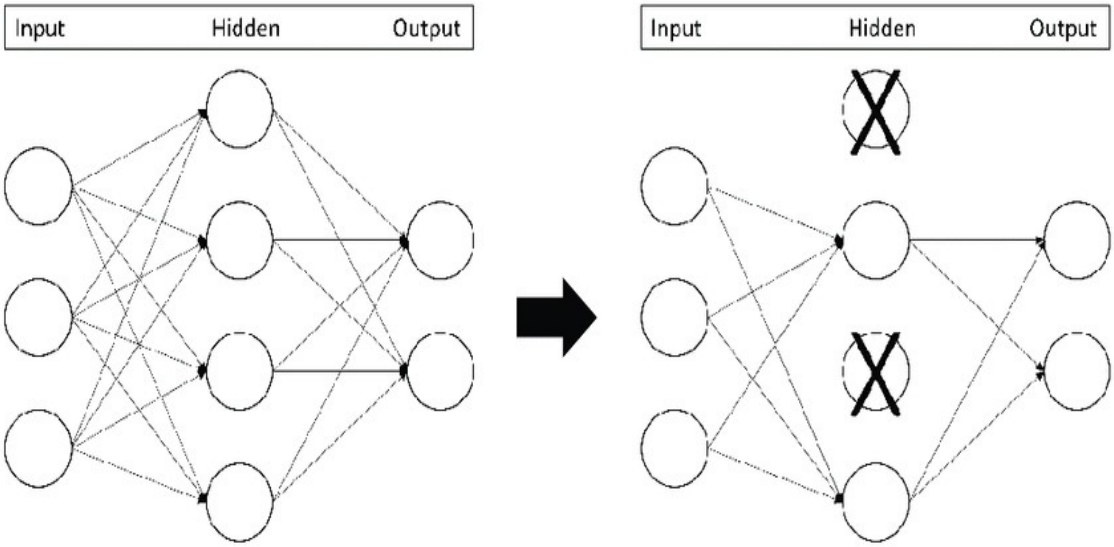
https://towardsai.net/p/l/introduction-to-pooling-layers-in-cnn

! Although it is technically optional, pooling layers are essential in almost every case.

4) Dropout Layer :

⟶ This layer drops out a fraction of neurons from the previous layer, reducing the number of units, to make the network learn more robust features.

⟶ It is used in deep CNNs, often with complex architectures.

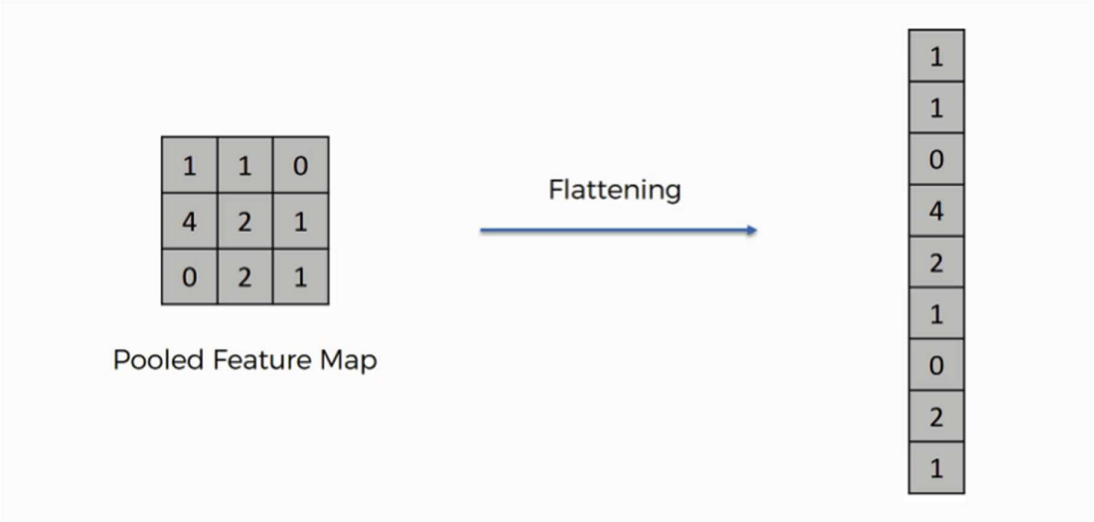


https://www.researchgate.net/figure/Example-of-dropout-layer-probability-of-50-appears-on-the-right_fig8_333411007

5) Flatten Layer :

⟶ This layer flattens the multi-dimensional data from convolutional blocks to one-dimensional vectors, preparing it for fully connected layers.

⟶ This one-dimensional vector suitable as input for fully connected layers, which typically work with flat data for classification or regression tasks.



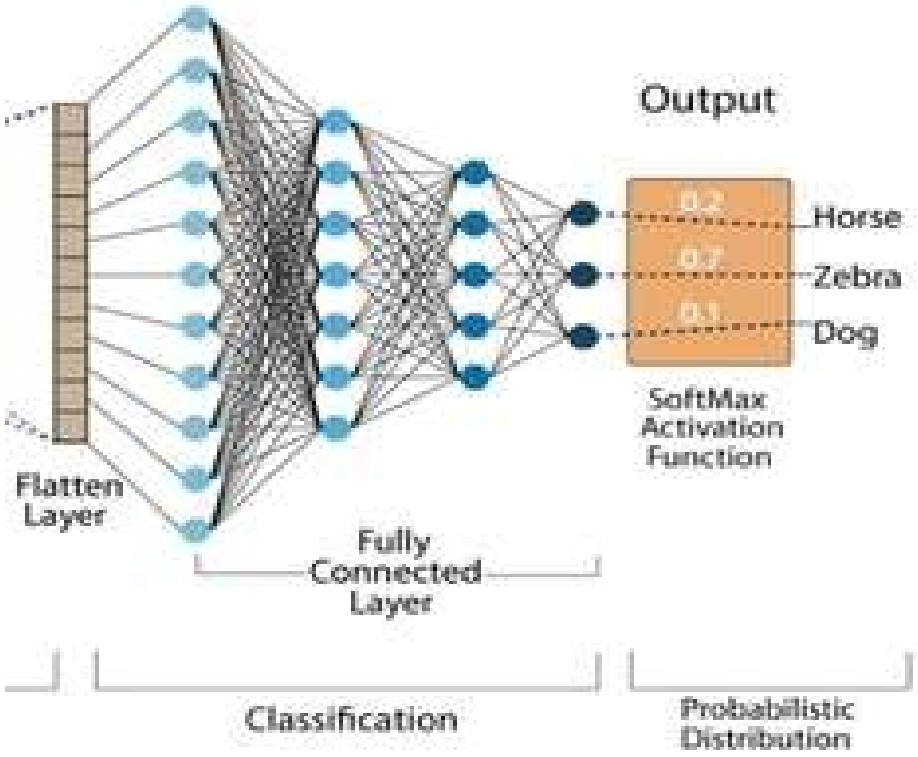
https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening

6) Fully Connected Layers :

⟶ The actual classification or prediction happens in these layers.

⟶ The normal fully-connected layers takes input from the Flatten layer and carries out the classification/prediction task.

⟶ The output layer, has the appropriate function like Softmax, Sigmoid etc. based on the task, which is usually multi-class or binary classification.

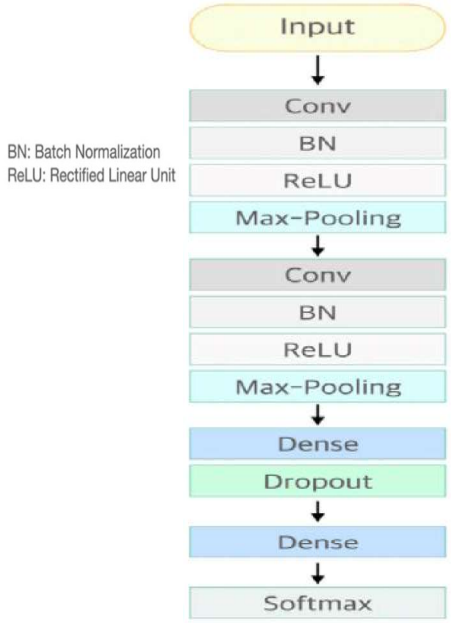


In a manner of understanding,

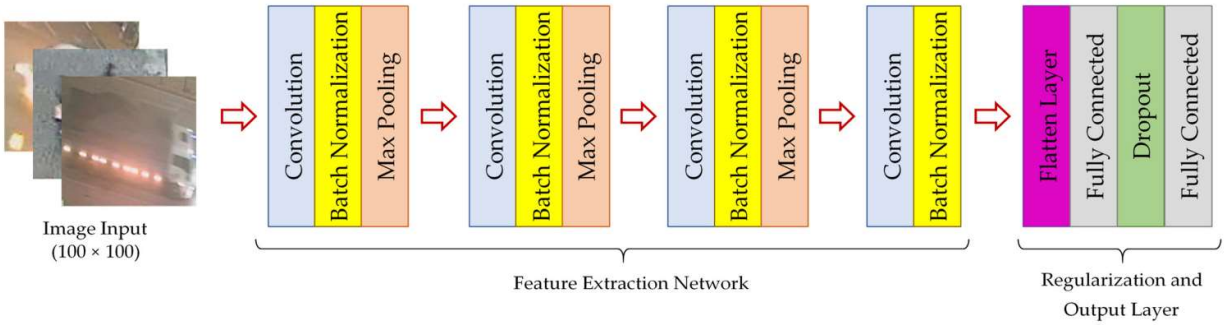
the fully connected layers can be interpreted as separate neural network for which the input is the feature maps generated by the convolutional portion of the network.

We combine all the parts sequentially to build a complete convolutional neural network.

Multiple convolution blocks, which will extract features, then a set of fully connected (Dense) layers which will use those features to make classification.

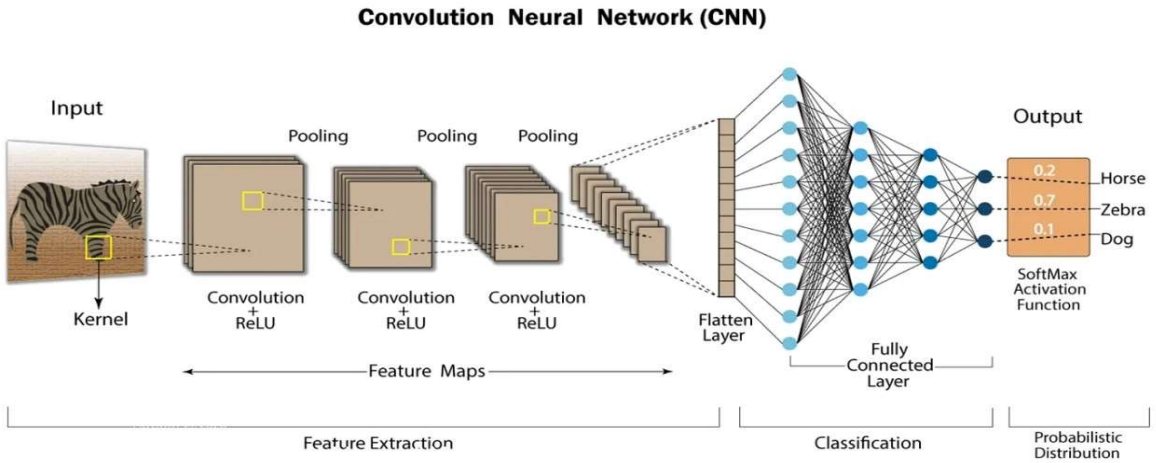


Some examples :



https://www.mdpi.com/2071-1050/15/23/16292

For classifying images of horses, zebras and dogs :



https://www.linkedin.com/pulse/what-convolutional-neural-network-cnn-deep-learning-nafiz-shahriar/

https://www.researchgate.net/figure/Functional-structure-of-the-proposed-convolutional-neural-network-architecture-Conv_fig2_350284279

Thank You !

If you found this article helpful, please do leave a like and comment any feedback you feel I could use.

While I’m working on the next project, I’ll be over at :

www.linkedin.com/in/shivang-kainthola-2835151b9/

shivang.kainthola64@gmail.com

https://shivangkainthola28.medium.com/

https://github.com/HeadHunter28