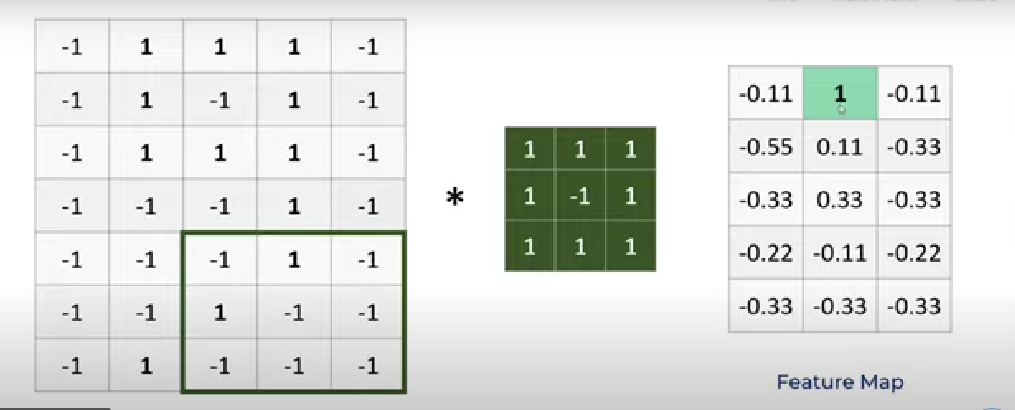
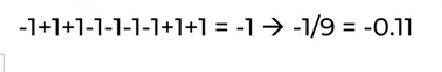
**CONVOLUTION NEURAL NETWORK**

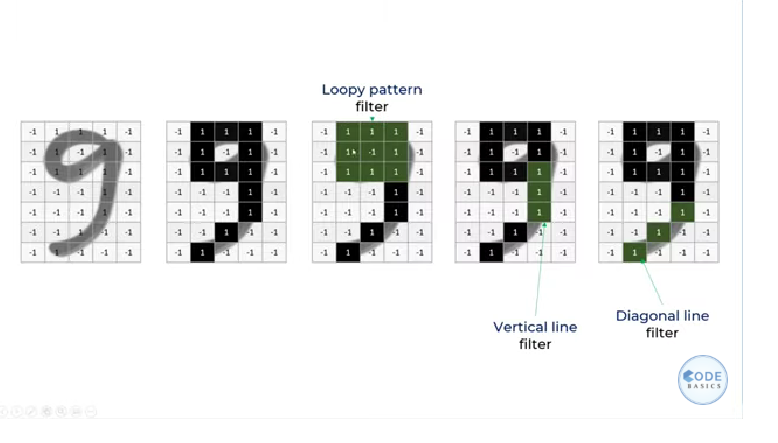
Traditional neural network will not efficiently working on a image EX: to detect a flying bird traditional approach will not work, the basic binary classification will not work out for this, CNN will do it in more efficiently by doing the following things.

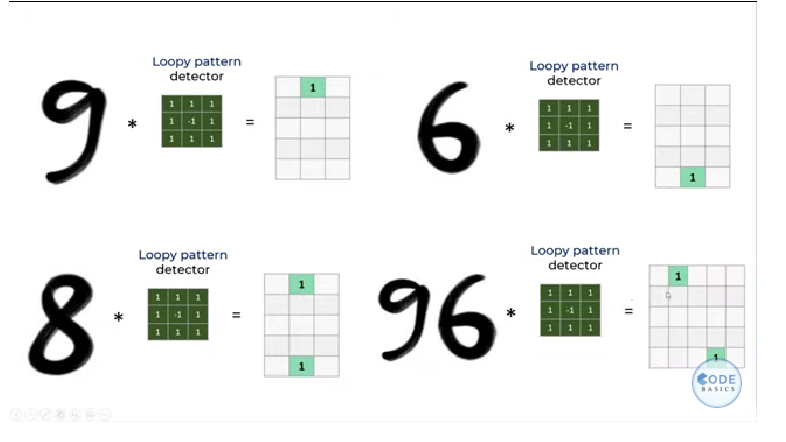
In convolution neural network we will take some dimension of a pattern from large image and multiply those dimension patter with the pattern that an trained model already have, In the below image we will do digit recognition, dark green image is the one of the 3X3 dimension pattern of image number 9, and that is multiplied with the pattern which model already have (selected white 3X3).

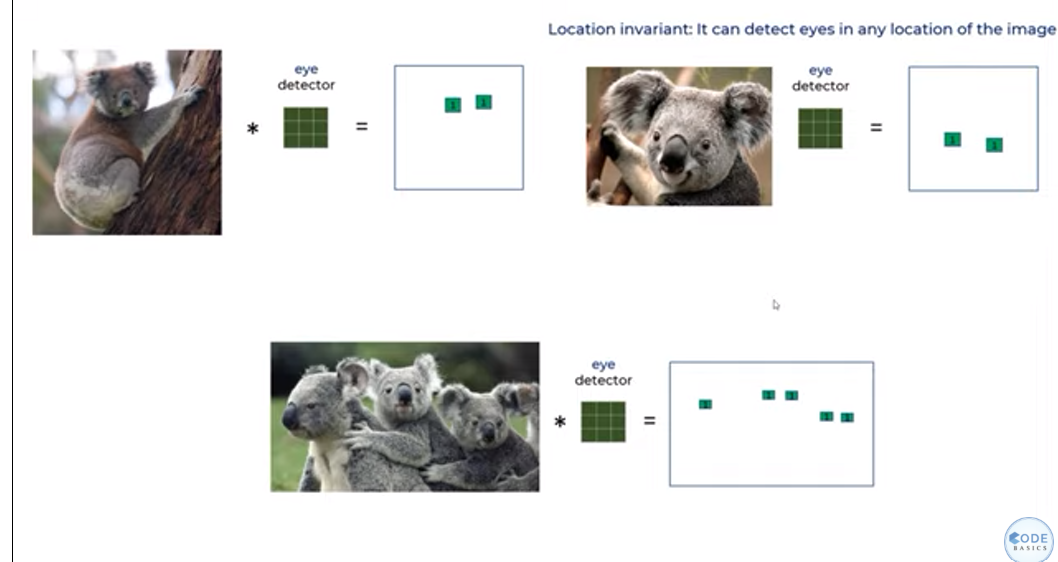
When an exact match is found the calculation will return 1, means it will be the part of image number 9, any number which is closet to 1 will be declared as part of the 9

****

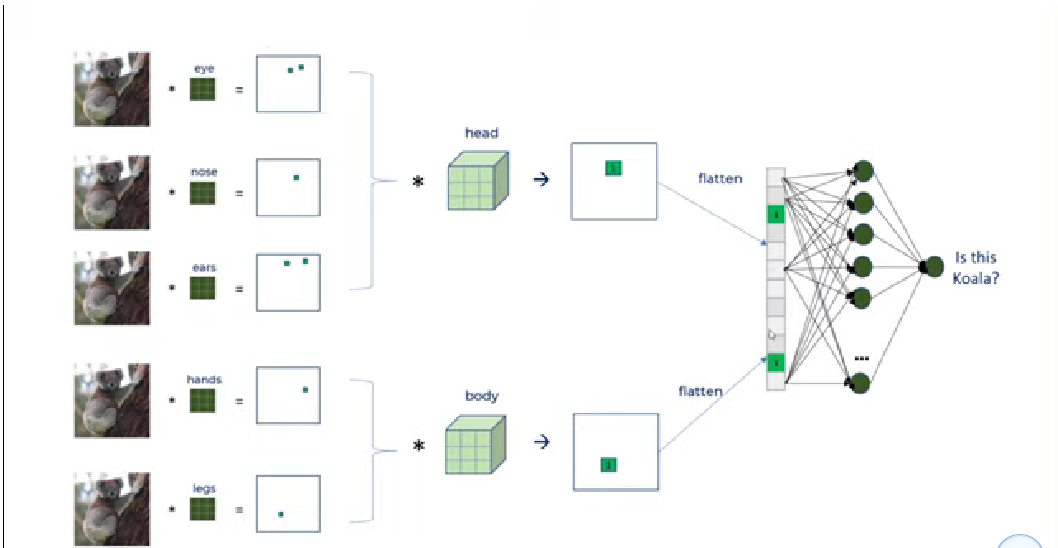
This 3X3 will be calculated like the below image add,sub all the number / sum of numbers taken to add,sub, This process is called **feature map.** Many no of feature map will be done to find matched pattern







The main feature of neural network is it will connect all the small piece of recognized pattern into a single one to match full exact image. The input will not be in 1,2 dimension in all time, it might be in 3D too, in 3D image also neural network finds the pattern and that will be again flattened to 2D array



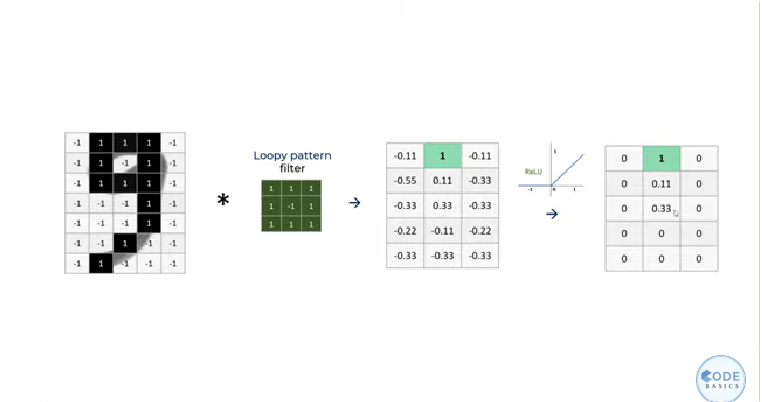
Steps on convolution neural network

1. Feature extraction – find all similar patterns, Back prorogation also will be done to do feature extraction
2. Classification – Which is done using densely connected neural network

**ReLu**

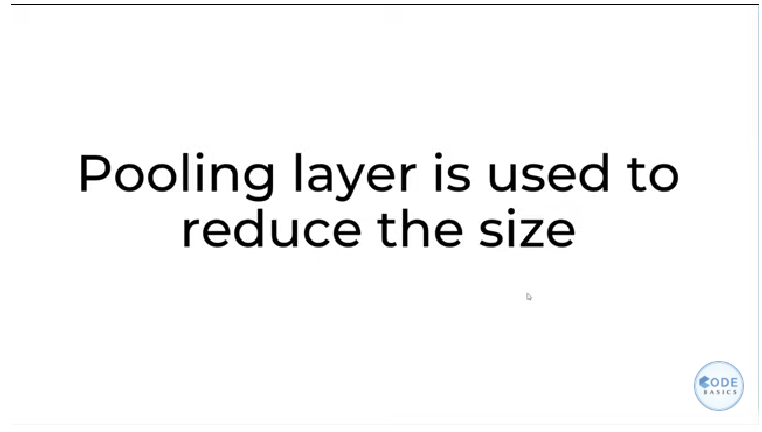


ReLu will be a activation function that removes all the negative pattern match values to train the model in non linear way, below image does that

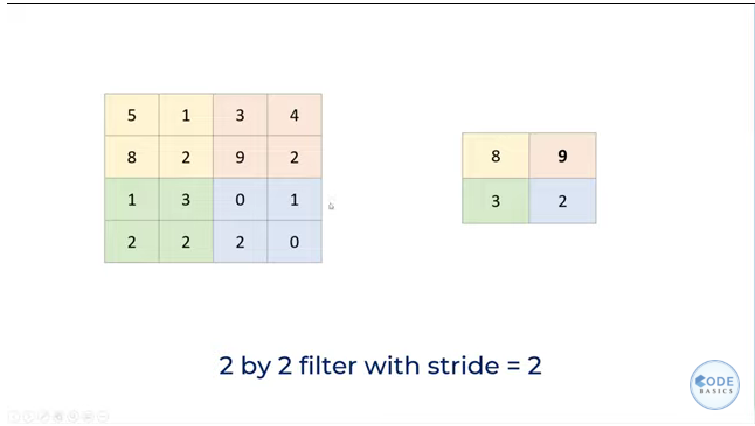


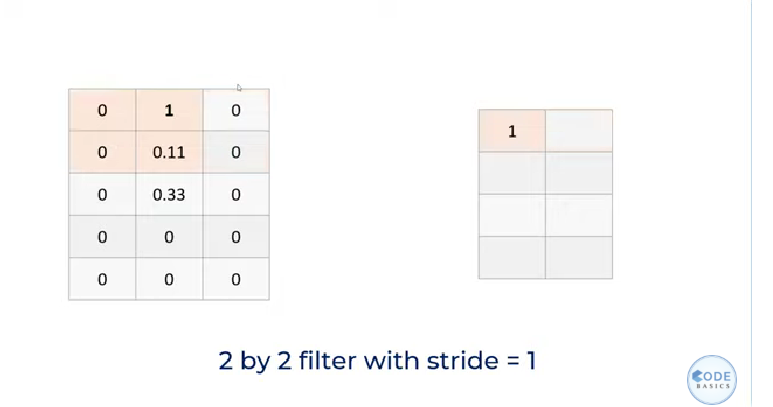
The no of process of this feature mapping will become Billon, trillion for large or more complex image, to do this more efficient way **pooling** are used here

1. Pooling

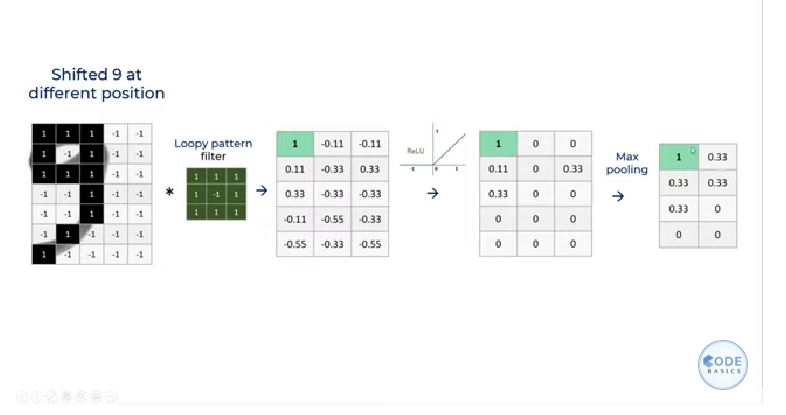


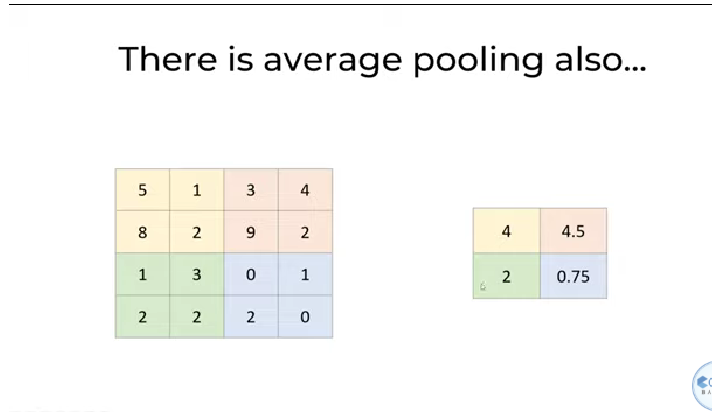
Pooling the process of taking the large number from some dimensions, Below image we did it.

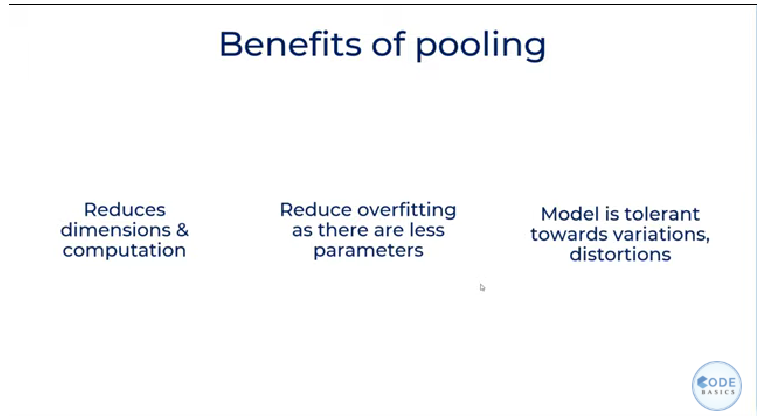




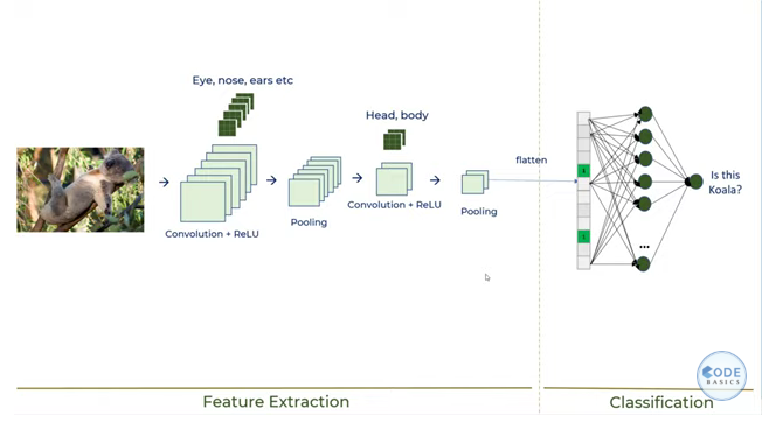
We will do this pooling to the ReLu data, so the below image has the max pollings from ReLu data



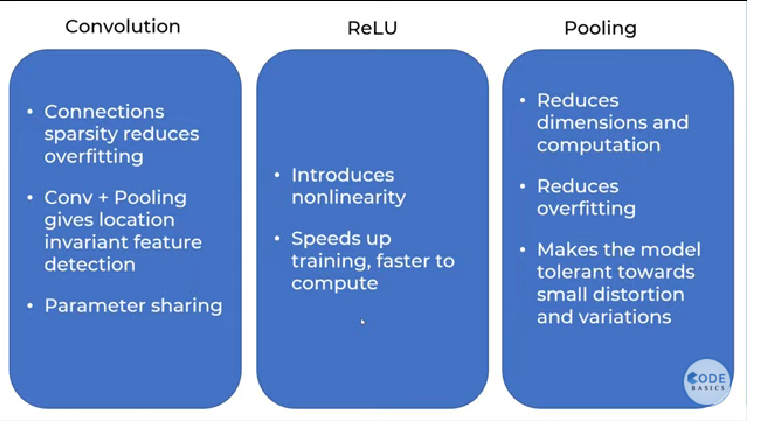




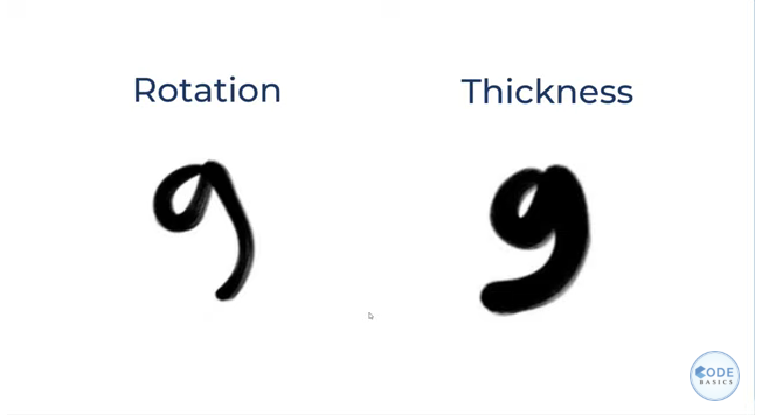
Complete CNN looks like the below image

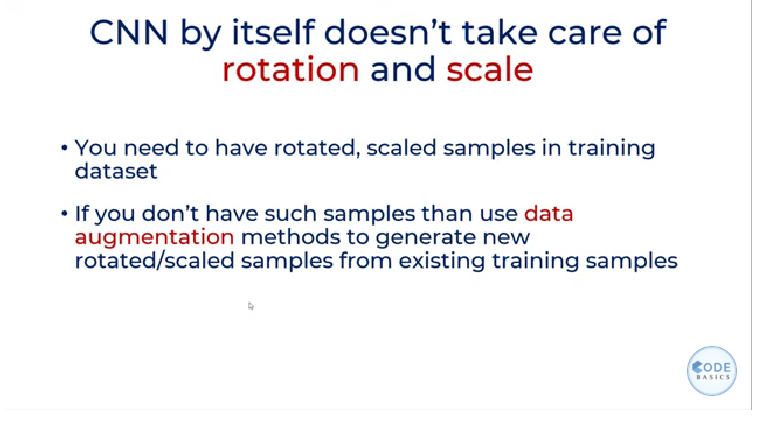


**BENEFITS OF THIS THREE**

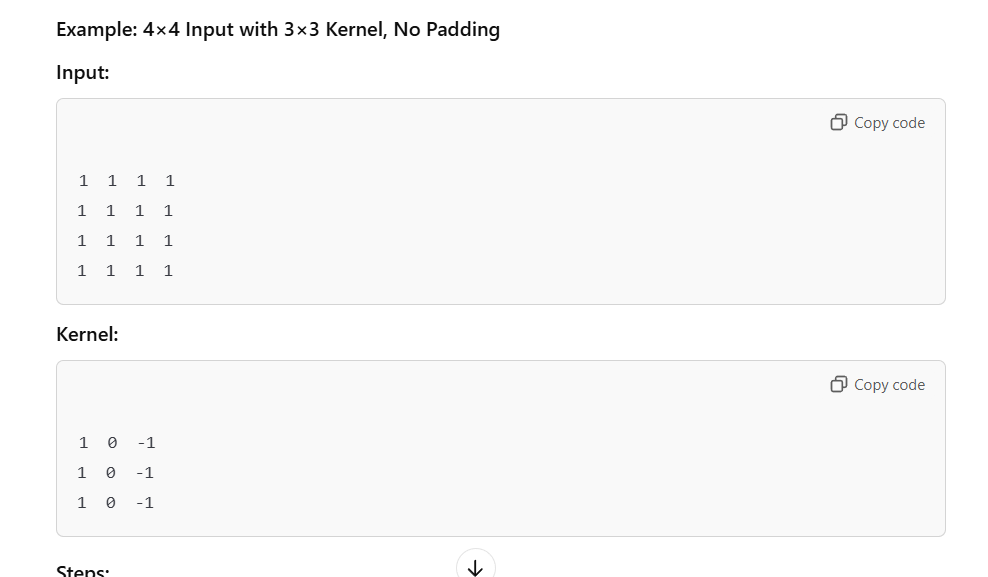


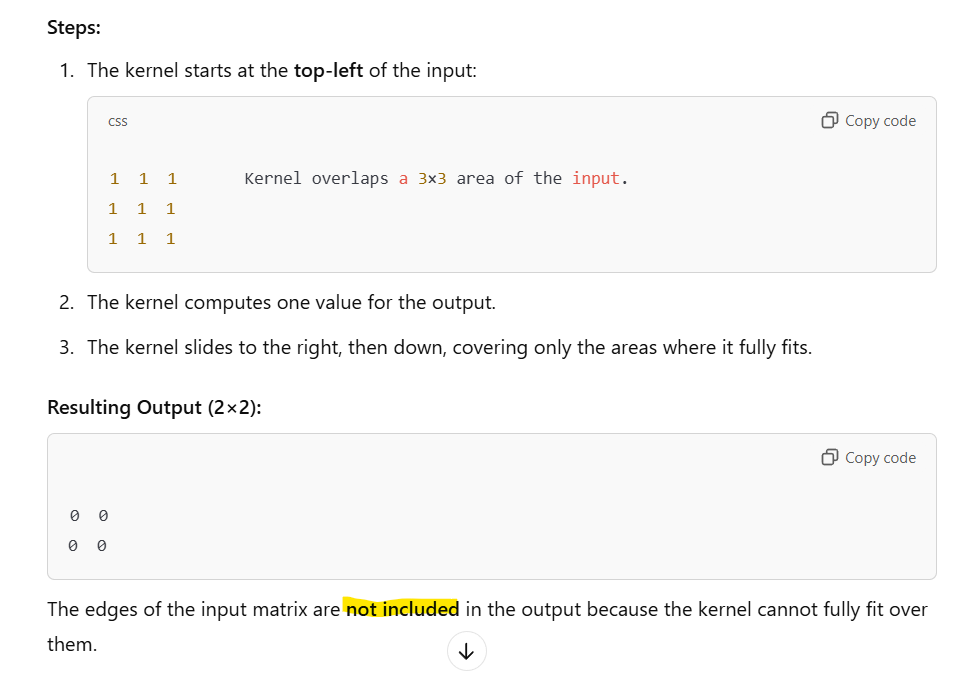
**DISADVANTAGES**

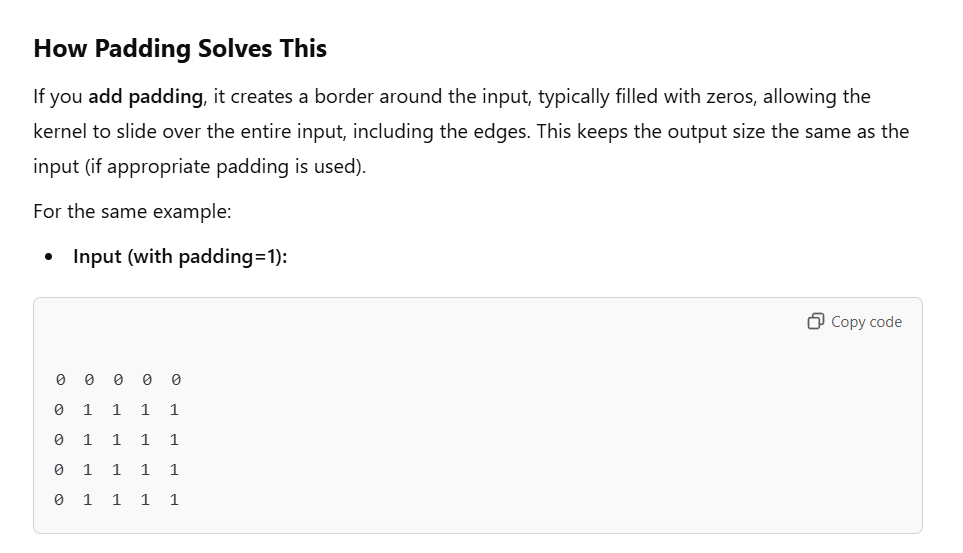


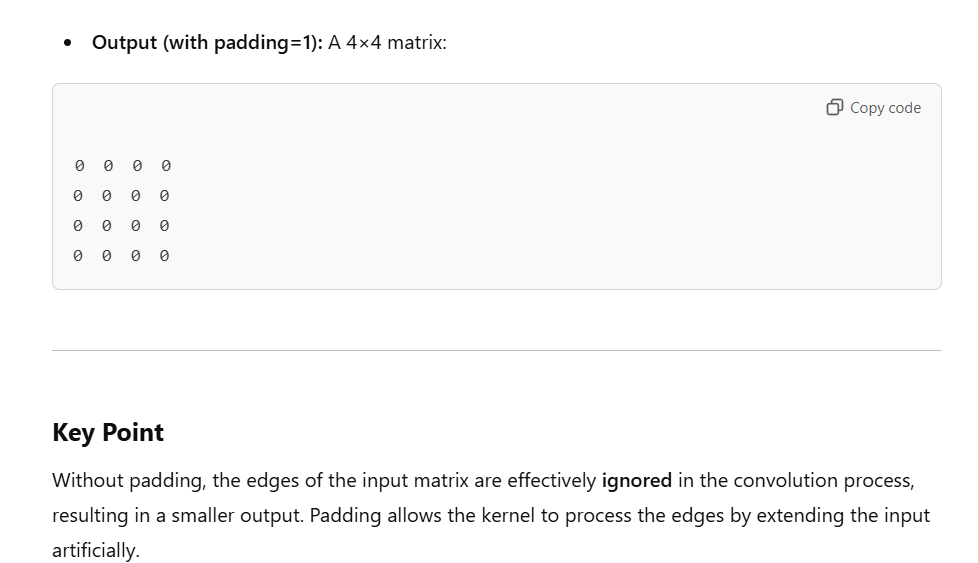


**PADDING (used in code to create CNN)**

****

****

****

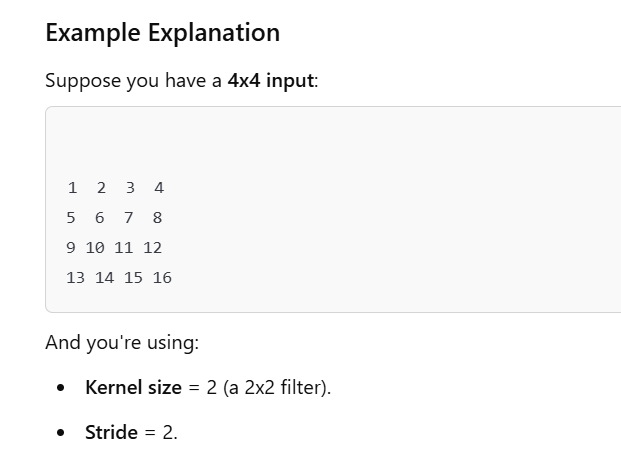
****

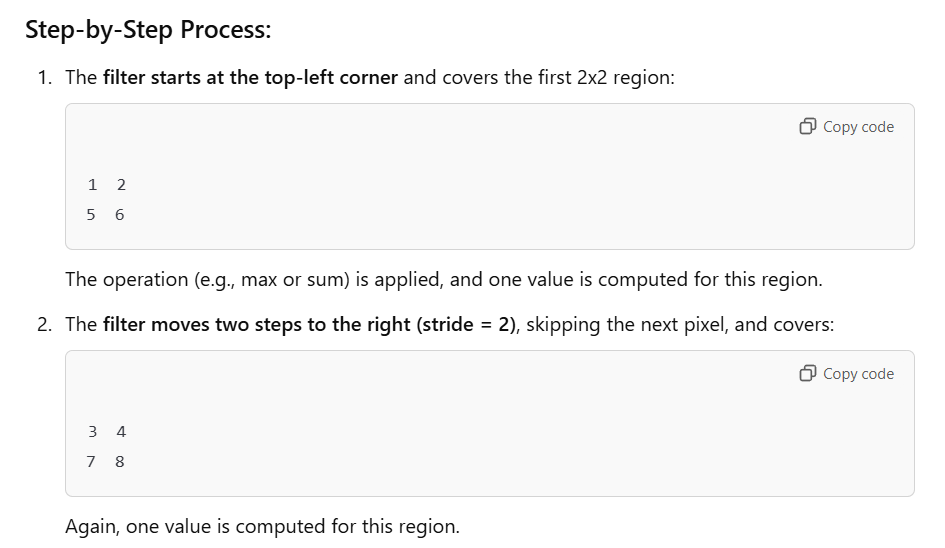
**DIFFERENCE BETWEEN USING KERNAL ON Conv2d and Maxpool2d (used in code to create CNN)**

* **Kernel Size**: The dimensions of the filter used in operations like Conv2d and MaxPool2d. The kernel is applied to a small portion of the input to compute a value:
  + In **Conv2d**, the kernel performs a convolution operation to extract features (like edges, textures, or patterns).
  + In **MaxPool2d**, the kernel selects the maximum value in its region, reducing the spatial size and retaining the most important features.

**STRIDE (used in code to create CNN)**

When **stride = 2**, the filter (or kernel) moves **two steps at a time** along the input. This means some pixels are skipped during the operation, which effectively reduces the spatial dimensions of the output.

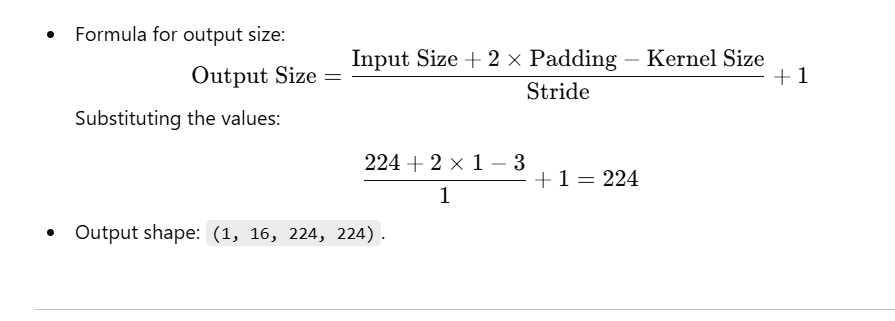




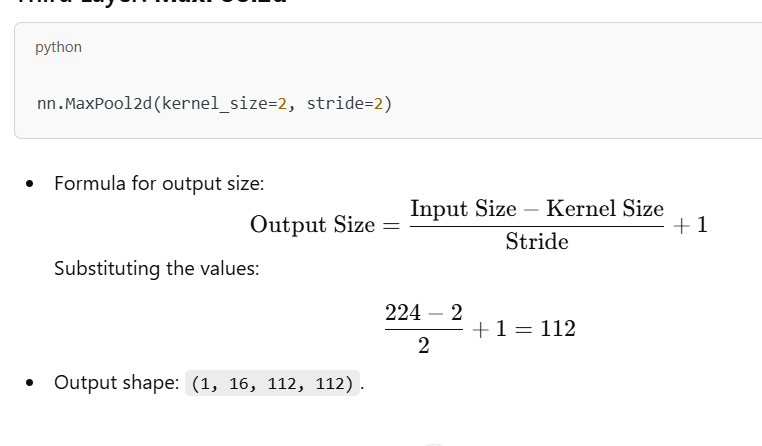
**Output size calculation of Conv2d and MaxPool2d (used in code to create CNN)**



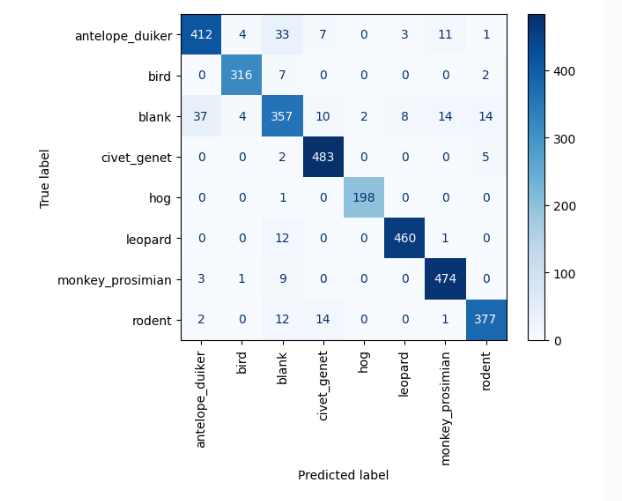
Output size of this layer can be calculated by the following way



For maxPool2d also it uses this calculation, but it doesn’t have in and out channels so it just do the following thing



**CNN model prediction program output:**

****

**What is a Confusion Matrix?**

A confusion matrix summarizes the performance of a classification model by showing the counts of correct and incorrect predictions for each class.

* **Rows** represent the **true labels** (actual classes in your dataset).
* **Columns** represent the **predicted labels** (what your model predicted for the inputs).

**Structure of the Matrix**

For this specific confusion matrix:

* Each cell represents the count of predictions for a combination of **true label** (row) and **predicted label** (column).
* **Diagonal cells** (top-left to bottom-right) represent the correctly predicted samples (true positives for each class).
* **Off-diagonal cells** show misclassifications—where the model predicted the wrong class.

**Understanding the Matrix**

Let’s analyze the rows and columns in your matrix:

1. **Row "antelope\_duiker":**
   * **True label:** The actual class is antelope\_duiker.
   * Diagonal cell (412): The model correctly predicted 412 images as antelope\_duiker.
   * Off-diagonal cells:
     + Predicted as blank (33 times).
     + Predicted as rodent (11 times), and so on.
2. **Row "bird":**
   * **True label:** The actual class is bird.
   * Diagonal cell (316): The model correctly predicted 316 images as bird.
   * Off-diagonal cells:
     + Predicted as blank (7 times).
     + Predicted as rodent (2 times), and so on.
3. **Row "civet\_genet":**
   * **True label:** The actual class is civet\_genet.
   * Diagonal cell (483): The model correctly predicted 483 images as civet\_genet.
   * Very few misclassifications.