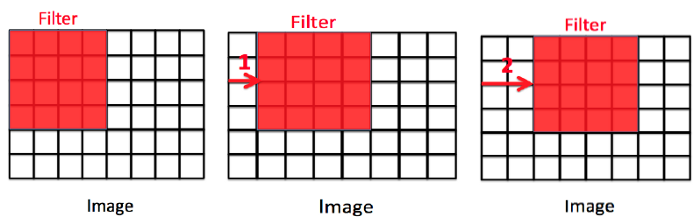
LAB-07

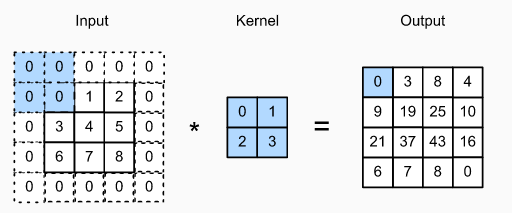
**Ques-2) What is Stride, Padding & Pooling? Explain with an example.**

**Strides**

When the array is created, the pixels are shifted over to the input matrix. The number of pixels turning to the input matrix is known as the strides. When the number of strides is 1, we move the filters to 1 pixel at a time. Similarly, when the number of strides is 2, we carry the filters to 2 pixels, and so on. They are essential because they control the convolution of the filter against the input, i.e., Strides are responsible for regulating the features that could be missed while flattening the image. They denote the number of steps we are moving in each convolution. The following figure shows how the convolution would work.

  
**Padding**

The padding plays a vital role in creating CNN. After the convolution operation, the original size of the image is shrunk. Also, in the image classification task, there are multiple convolution layers after which our original image is shrunk after every step, which we don’t want. Secondly, when the kernel moves over the original image, it passes through the middle layer more times than the edge layers, due to which there occurs an overlap.



**Pooling**

The pooling layer is another building block of a CNN and plays a vital role in pre-processing an image. In the pre-process, the image size shrinks by reducing the number of parameters if the image is too large. When the picture is shrunk, the pixel density is also reduced, the downscaled image is obtained from the previous layers. Basically, its function is to progressively reduce the spatial size of the image to reduce the network complexity and computational cost.

Reference link - <https://www.codingninjas.com/codestudio/library/convolution-layer-padding-stride-and-pooling-in-cnn>

**Ques-4) What is Stride, Padding & Pooling? Explain with an example.**

When machine learning algorithms are constructed, they leverage a sample dataset to train the model. However, when the model trains for too long on sample data or when the model is too complex, it can start to learn the “noise,” or irrelevant information, within the dataset. When the model memorizes the noise and fits too closely to the training set, the model becomes “overfitted,” and it is unable to generalize well to new data. If a model cannot generalize well to new data, then it will not be able to perform the classification or prediction tasks that it was intended for. Low error rates and a high variance are good indicators of overfitting.

**How to avoid overfitting**

* **Train with more data:** Expanding the training set to include more data can increase the accuracy of the model by providing more opportunities to parse out the dominant relationship among the input and output variables.
* **Data augmentation:** While it is better to inject clean, relevant data into your training data, sometimes noisy data is added to make a model more stable. However, this method should be done sparingly.
* **Feature selection:** When you build a model, you’ll have a number of parameters or features that are used to predict a given outcome, but many times, these features can be redundant to others. Feature selection is the process of identifying the most important ones within the training data and then eliminating the irrelevant or redundant ones.
* **Regularization:** Regularization applies a “penalty” to the input parameters with the larger coefficients, which subsequently limits the amount of variance in the model.  While there are a number of regularization methods, such as L1 regularization, Lasso regularization, and dropout, they all seek to identify and reduce the noise within the data.
* **Ensemble methods:** Ensemble learning methods are made up of a set of classifiers—e.g. decision trees—and their predictions are aggregated to identify the most popular result. The most well-known ensemble methods are bagging and boosting.