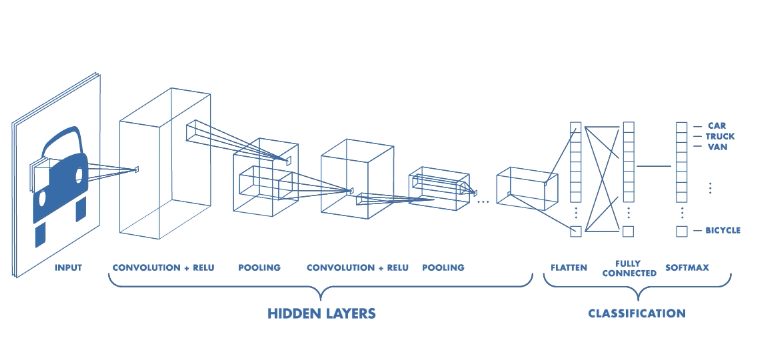
**CIFAR-10 Dataset Image Classification**

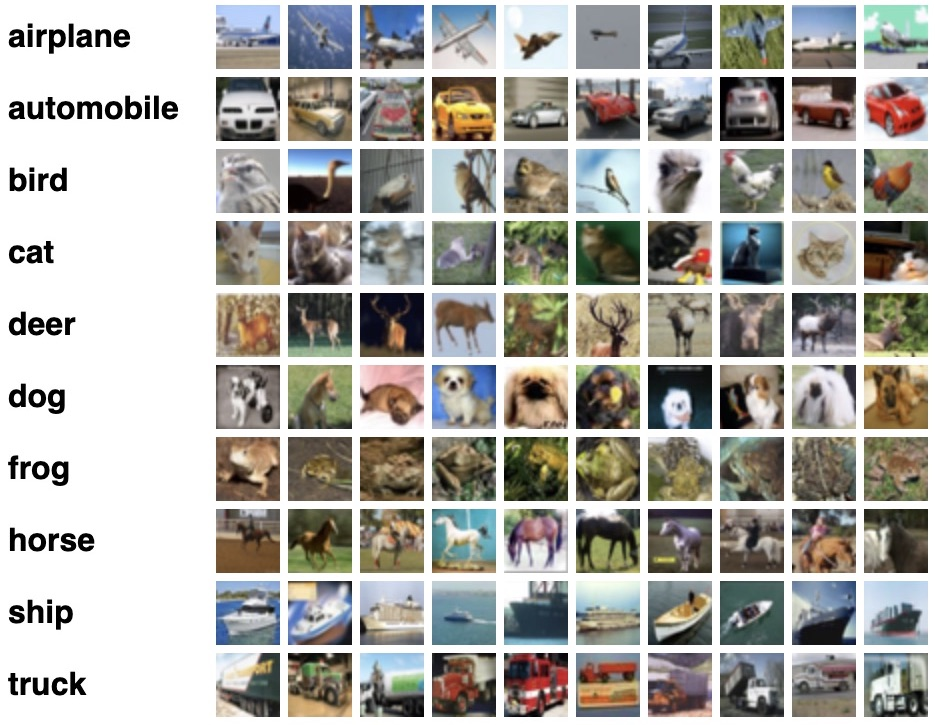
Neural Networks are programmable patterns that aid in the solution of complex issues and produce the best possible results. As we all know, Deep Learning is a step forward of Machine Learning, and it aids in the training of Neural Networks for obtaining answers to unresolved issues and/or enhancing the solution!

I describe a Deep Learning Model utilizing the CIFAR-10 dataset in this post. The dataset is frequently used in Deep Learning to test Image Classification algorithms. It has 60,000 color photos divided into ten categories. The images are 32x32 pixels in size, and the dataset contains 50,000 training and 10,000 test images.

Traditional neural networks have achieved commendable image classification performance, but they are characterized by feature engineering, a time-consuming process that leads to poor generalization to test data. To extract underlying image information, learnable filters and pooling layers were used. To avoid overfitting and provide better accuracies in validation and testing, dropout, regularization, and variation in convolution methods were used. A deeper network resulted in improved test accuracy and less overfitting.

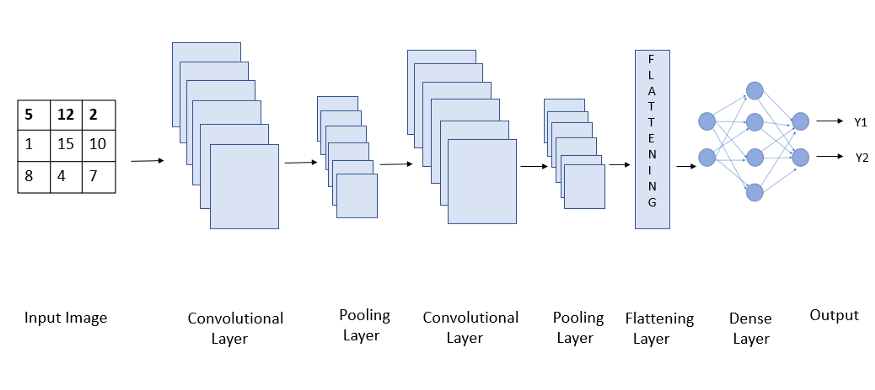


Each of the 10000 photos in the dataset is separated into five training batches and one test batch. The test batch contains exactly 1000 photographs from each class, chosen at random. The remaining photographs are distributed in training batches in a random order, however some batches may contain more images from one class than others. The training batches contain exactly 5000 photos from each class between them.



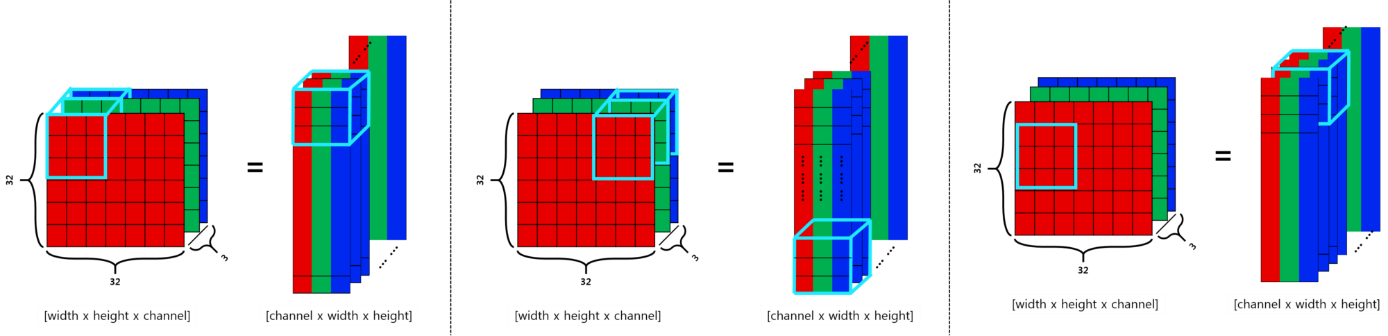
**Building CNN model**

A CNN model has three stages of operation. A convolutional layer extracts the features of the image/data in the first stage. A pooling layer decreases the image's dimensionality in the second stage, allowing modest alterations to have a huge impact on the model. Simply said, it keeps you from over-fitting. A flattening layer modifies our model in one dimension and feeds it to the fully linked dense layer in the third step. The image is then predicted by this dense layer. Multiple layers of convolutional and pooling layers make to a decent model.



There are two types of APIs that are commonly utilized while building a Neural Network model: Sequential API and Functional API. We can use the Sequential API to generate a model layer by layer and then add it to a sequential Class. The disadvantage of Sequential API is that it cannot be used to build a model with numerous input sources and outputs at separate locations. We use Functional API to get over this limitation. We can develop many input and output models using the Functional API. However, Sequential API is used in the majority of scenarios. For our CNN model, we'll use Sequential API.

I'll use a convolutional layer because I am utilizing a Convolutional Neural Network. Conv2D is the most commonly utilized and the layer we are employing. Convolution on two axes is referred to as Conv2D. The convolution is extended to three strata: Red, Green, and Blue. Conv1D is the other sort of convolutional layer. Conv1D is typically used for "texts," whereas Conv2D is typically used for "pictures."



**Conclusion**

I was able to achieve a 78 percent accuracy rate using the model. As a result, in this essay, I will explain how the Deep Learning project works. The parameters utilized in the Convolutional Layer and Pooling Layer of a Convolutional Neural Network are familiar to me. I'll need a dense layer and a dropout after extracting features in a CNN to use these features in picture recognition. Finally, I learn a little more about loss functions and the Adam optimizer.

My Dataset Link:

https://www.kaggle.com/fedesoriano/cifar10-python-in-csv