VIT AP

Course code: CSE3008

Course Title: Introduction to Machine Learning

Module 5: Introduction to Deep Learning

DEEP LEARNING

- Deep learning is a sub field of Machine Learning that very closely tries to mimic human brain's working using neurons.

- These techniques focus on building Artificial Neural Networks (ANN) using several hidden layers.

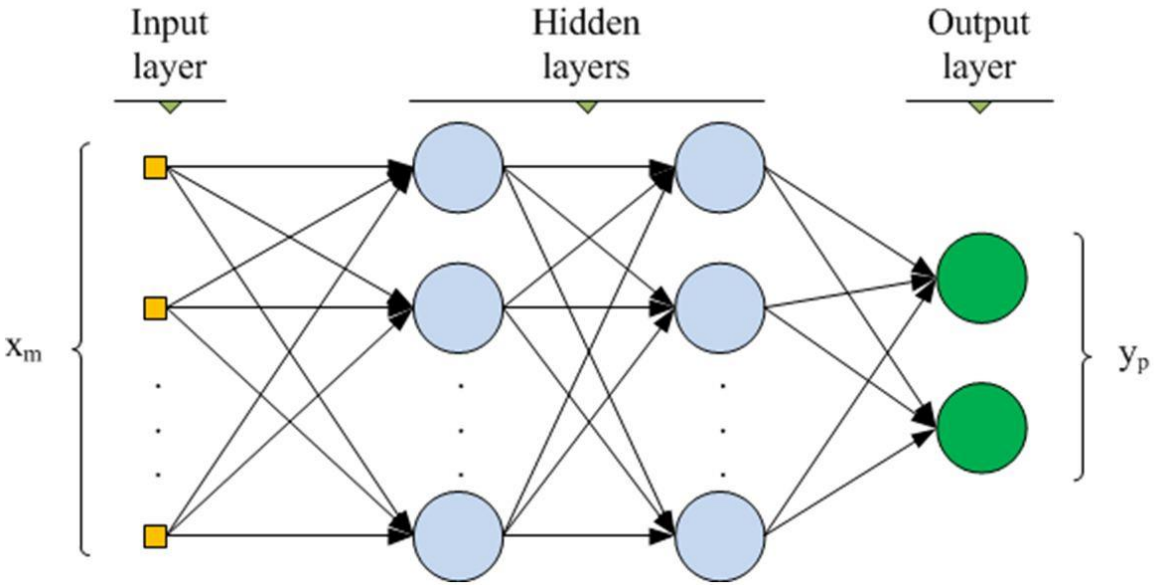
- There are variety of deep learning networks such as Multilayer Perceptron (MLP), Autoencoders (AE), Convolution Neural Network (CNN), Recurrent Neural Network (RNN).

Why DEEP LEARNING is Growing?

- Processing power needed for Deep learning is readily becoming available using GPUs, Distributed Computing and powerful CPUs.
- Moreover, as the data amount grows, Deep Learning models seem to outperform Machine Learning models.
- Focus on customization and real time decisioning.
- Uncover hard to detect patterns (using traditional techniques). Find latent features (super variables) without significant manual feature engineering.

Building Blocks of Deep Learning

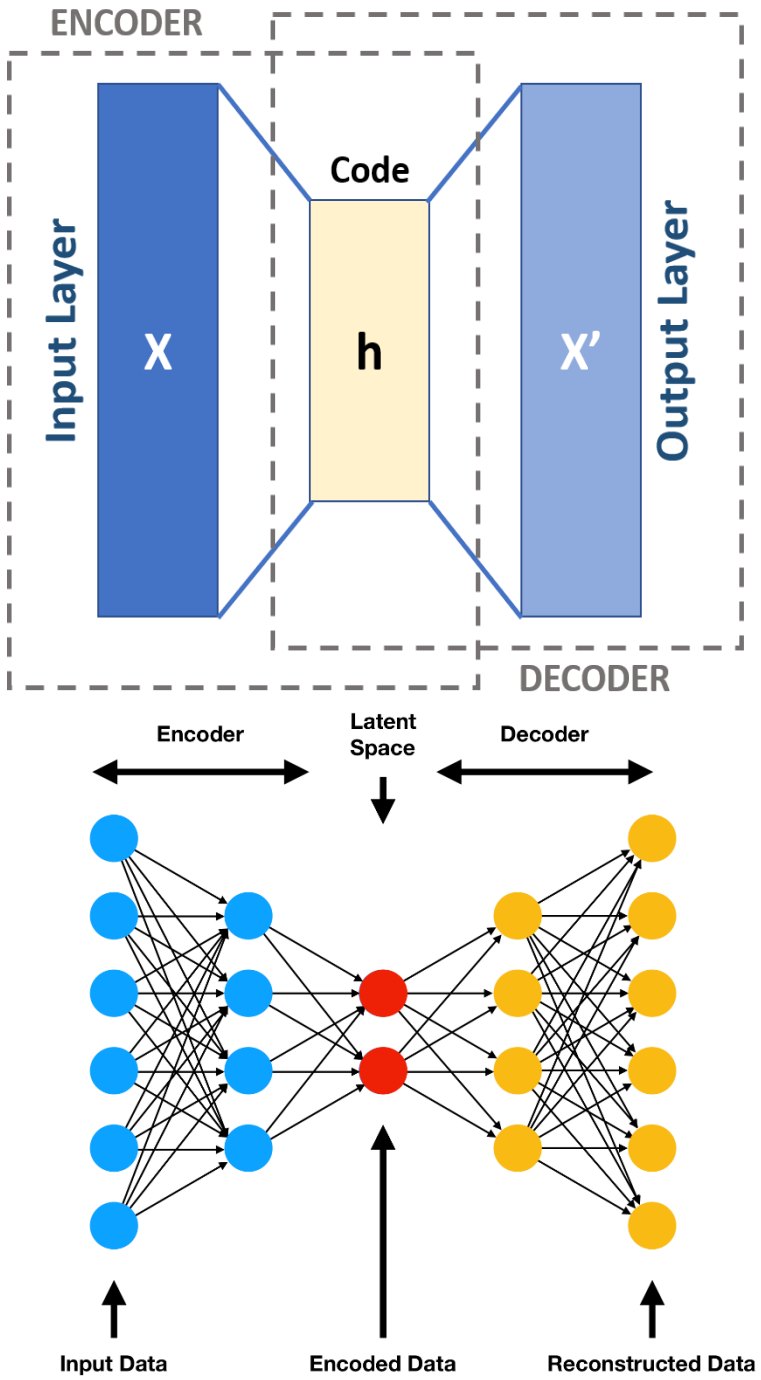
- **Multilayer Perceptron (MLP)**
  - These are the most basic networks and feed forward the inputs to create output.
  - They consist of an input layer and an output layer and many interconnected hidden layers and neurons between the input and the output layers.
  - They generally use some non linear activation function such as Relu or Tanh and compute the losses (the difference between the true output and computed output) such as Mean Square Error (MSE), Logloss.
  - This loss is backward propagated to adjust the weights and training to minimize the losses or make the models more accurate.



A (WX+B)

Autoencoders (AE)

- The concept of the AE is quite simple- the input vectors are used to compute the output vectors, but output vectors are same as the input vectors.
- Autoencoders are neural networks that learn to efficiently compress and encode data then learn to reconstruct the data back from the reduced encoded representation to a representation that is as close to the original input as possible.
- Therefore, autoencoders reduce the dimensionality of the input data i.e. reducing the number of features that describe input data.



Convolution Neural Network

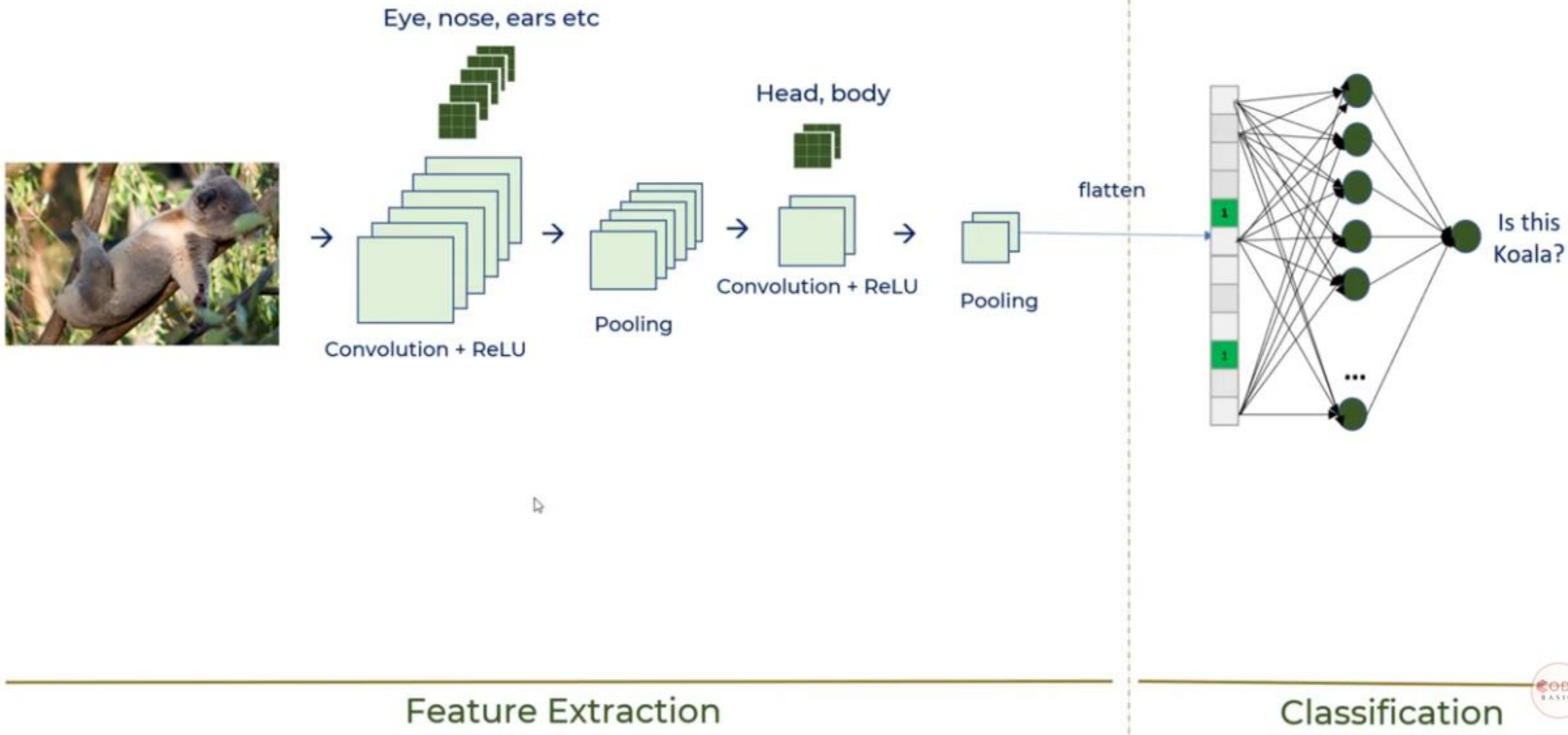
- Convolution Neural Networks (CNN) significantly enhances the capabilities of the feed forward network such as MLP by inserting convolution layers.
- They are particularly suitable for spatial data, object recognition and image analysis using multidimensional neurons structures.
- CNNs use convolutions (a linear operation) rather than matrix multiplication as in MLP
- Typically a CNN will have three stages- convolution stage, detector layer (non linear activator) and pooling layer

Convolution Neural Network

- **Convolution Layer-** The most important component in the CNN. The layer has Kernels (learnable filters) and the input x and y dimensions are convoluted (dot product) to generate feature map

- **Detector Layer-** The feature maps are passed to this stage using a not linear activation function such as ReLU activation function to accentuate the non linear components of the feature maps

- **Pooling Layer-** A pooling layer such as “max pooling” summarizes (sub-sampling) the responses from several inputs from the previous layer and serves to reduce the size of the spatial representation.


Eye, nose, ears etc
Head, body
Convolution + ReLU
Pooling
Convolution + ReLU
Pooling
flatten
Is this Koala?
Feature Extraction
Classification

Filters

- **Filters or Feature detectors** are like weight matrix which we multiply to the input image to generate the feature maps or the activation maps.

- **Feature detectors or filters** help identify different features present in an image like edges, vertical lines, horizontal lines, bends, etc.

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Image

Filter

1	0	1
0	1	0
1	0	1

Filter

4	0	0
0	0	0
0	0	0

- So, Convolution of Filter, reduces the image size

Filters

INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

WEIGHT

1	0	1
0	1	0
1	0	1

429

INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

WEIGHT

1	0	1
0	1	0
1	0	1

429	505	686	856

INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

WEIGHT

1	0	1
0	1	0
1	0	1

429	505	686	856
261	792	412	640

INPUT IMAGE

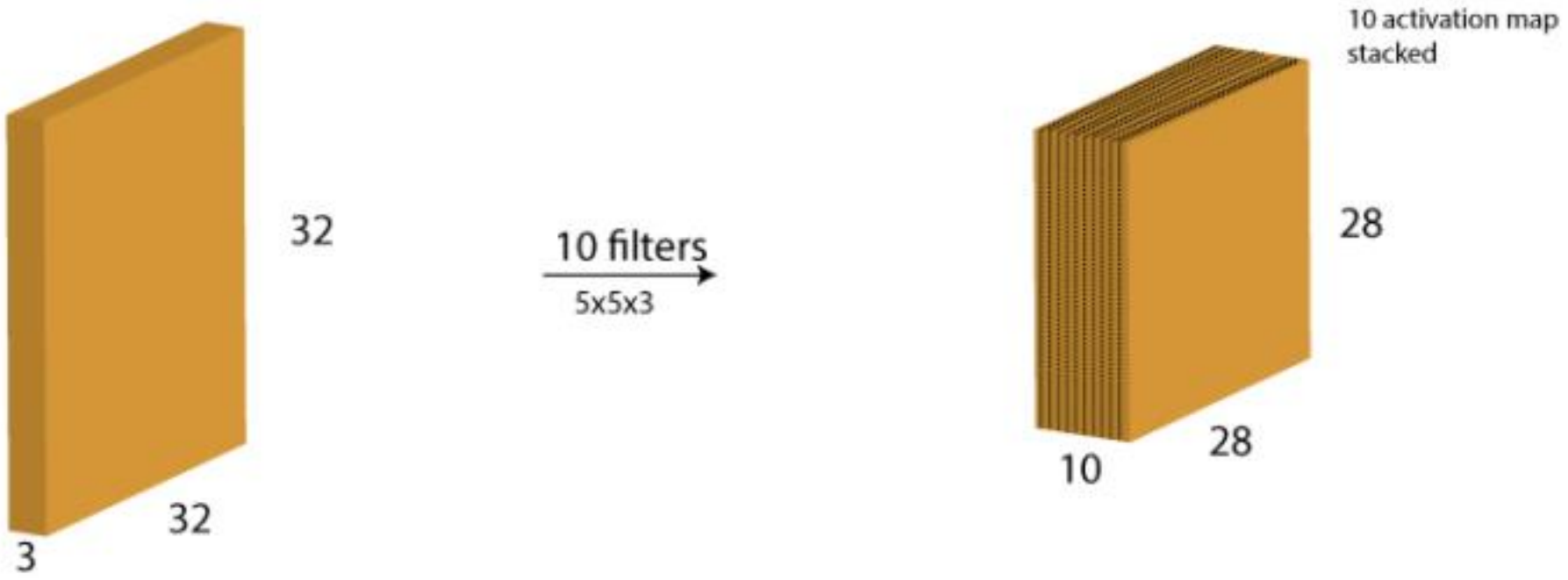
18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

WEIGHT

1	0	1
0	1	0
1	0	1

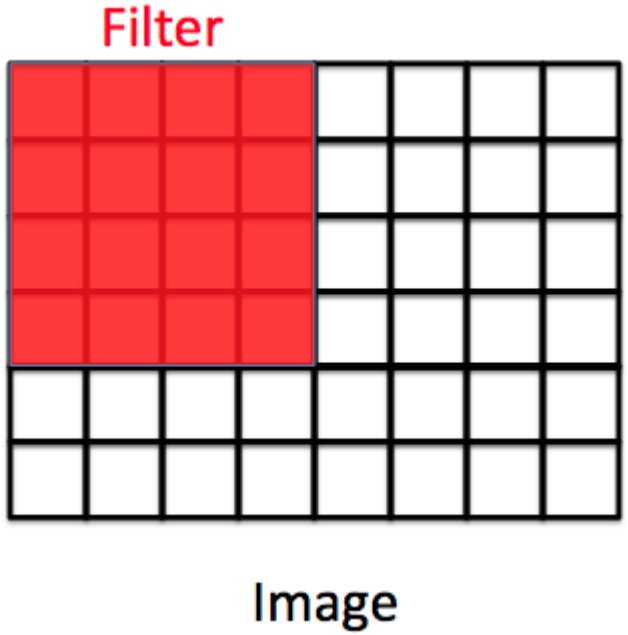
429	505	686	856
261	792	412	640
633			

Filters

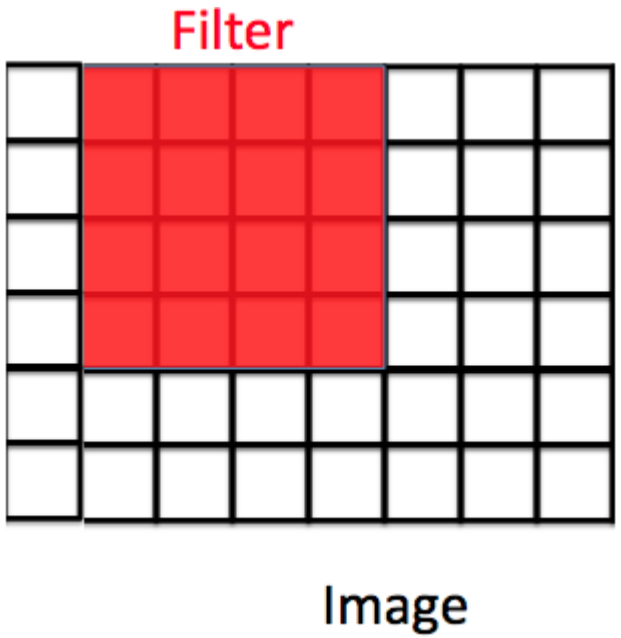


Filters

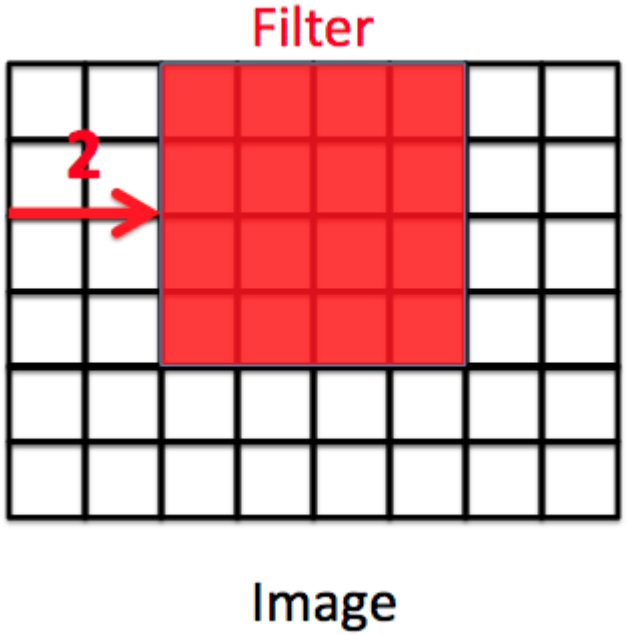
- **Stride:** defines how far the filter moves from one position to the next position by “stride



If stride =1, the filter will move one pixels.



If stride = 2, the filter will move two pixels.



Application of Edge Detection Filters

Convolution Neural Network

- **Pooling:** This is basically done to reduce the number of parameters to learn and prevents overfitting. Thus, it reduces the number of parameters to learn and the amount of computation performed in the network.

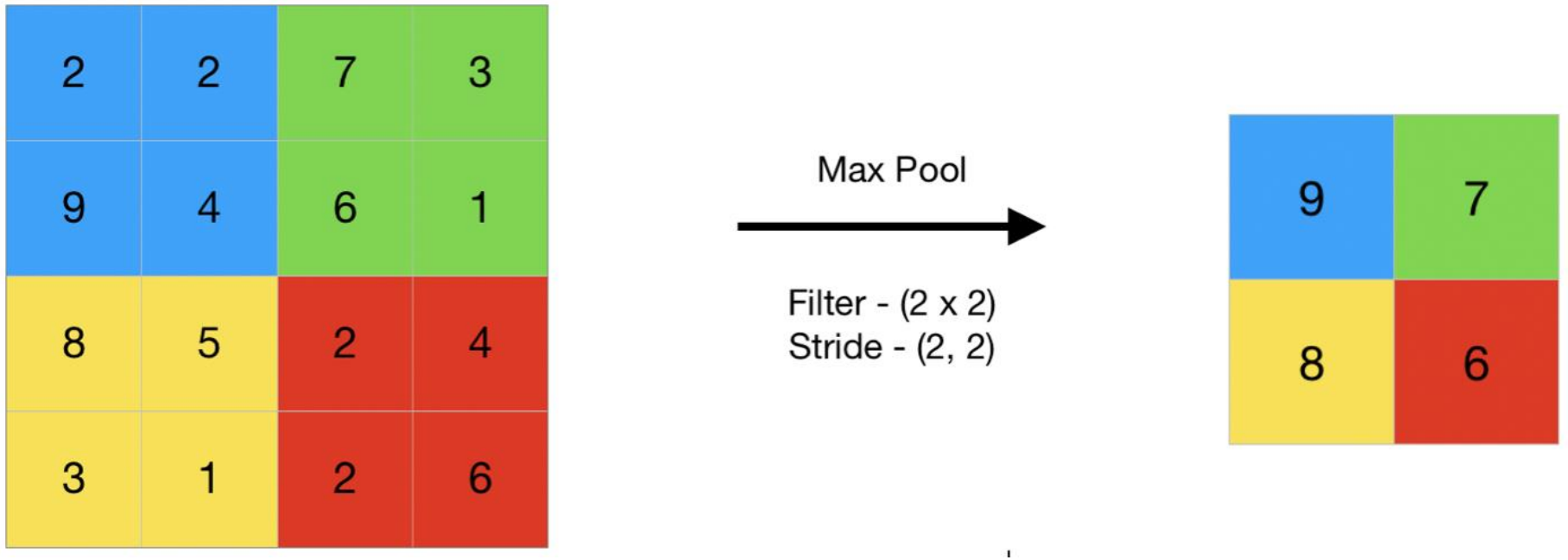
- Pooling is then applied over the feature maps for invariance to translation.

- Types of Pooling: **Max Pooling, Min Pooling** and **Average Pooling**

- Max pooling provides better performance compared to min or average pooling.

Convolution Neural Network

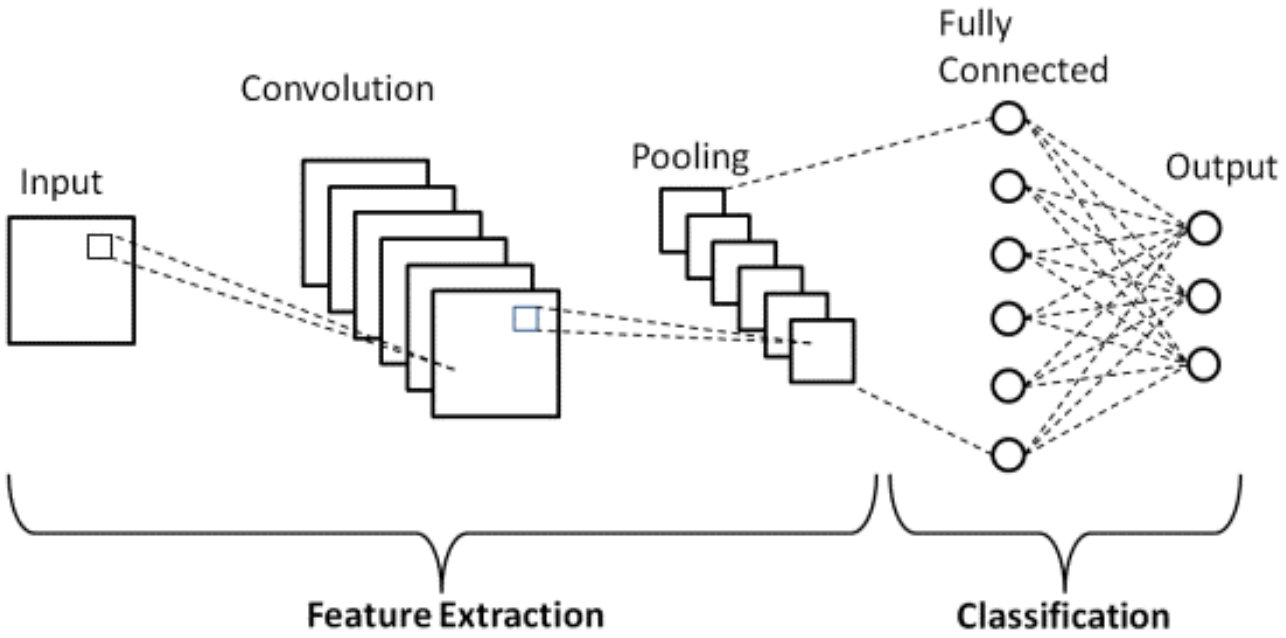
- **Max Pooling:**



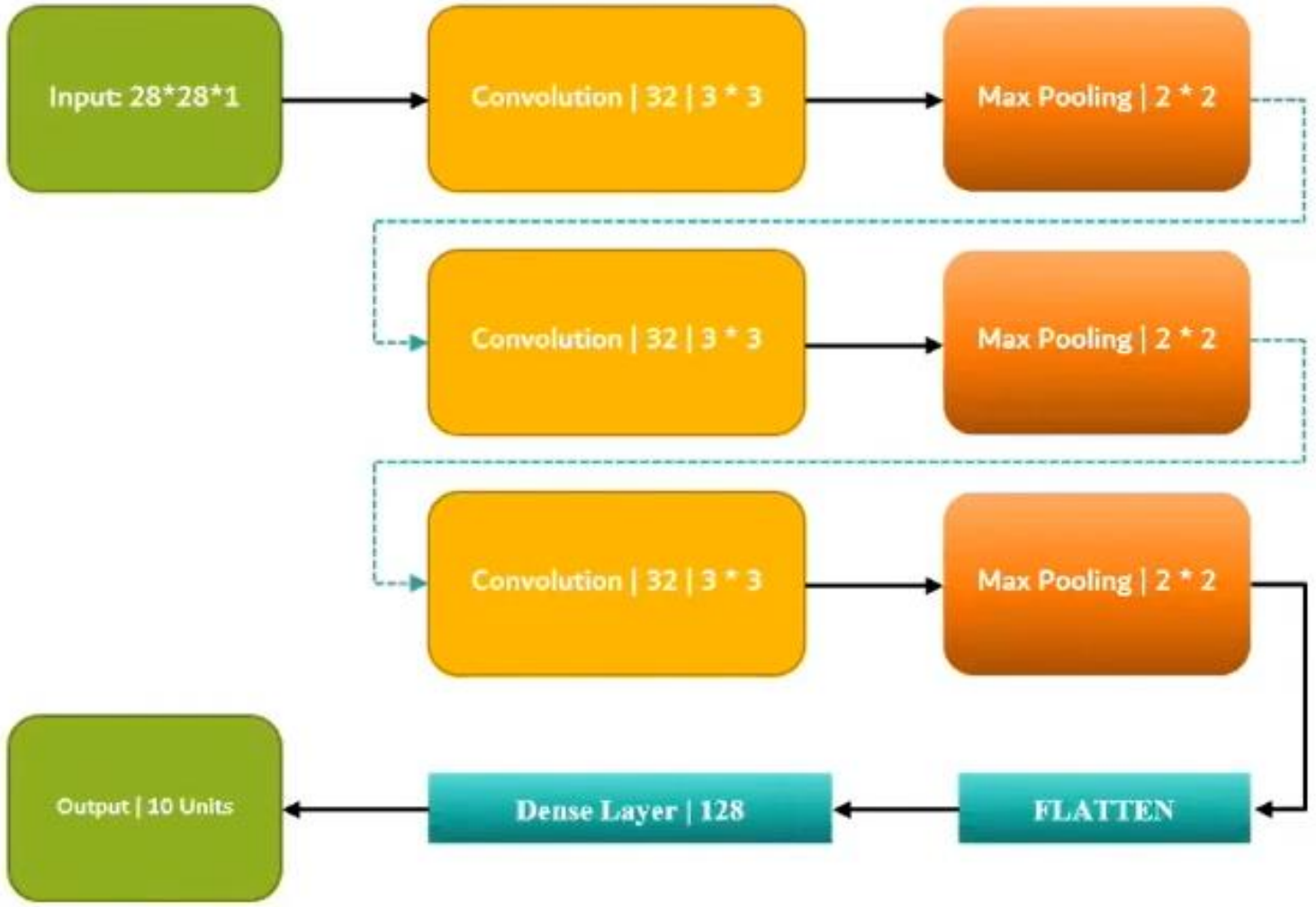
Fully Connected Layer

- This layer basically takes an input from its preceding layer and outputs an N dimensional vector where N is the number of classes.

- So, it is called flatten all the input and pass these flattened inputs to a deep neural network that outputs the class of the object.



Example: Convolution Neural Network



Convolution

- Connections sparsity reduces overfitting
- Conv + Pooling gives location invariant feature detection
- Parameter sharing

ReLU

- Introduces nonlinearity
- Speeds up training, faster to compute

Pooling

- Reduces dimensions and computation
- Reduces overfitting
- Makes the model tolerant towards small distortion and variations

Key points about Convolution layers and Filters

- **The depth of a filter in a CNN must match the depth of the input image**. The number of color channels in the filter must remain the same as the input image.
- **Different Conv2D filters are created for each of the three channels** for a color image.
- **Filters for each layer are randomly initialized based on either Normal or Gaussian distribution.**
- **Initial layers of a convolutional network extract lower-level features from the image, so use fewer filters.** As we build further deeper layers, we increase the number of filters to twice or thrice the size of the filter of the previous layer.
- **Filters of the deeper layers learn more features but are computationally very intensive**.

LeNet-5 CNN Architecture

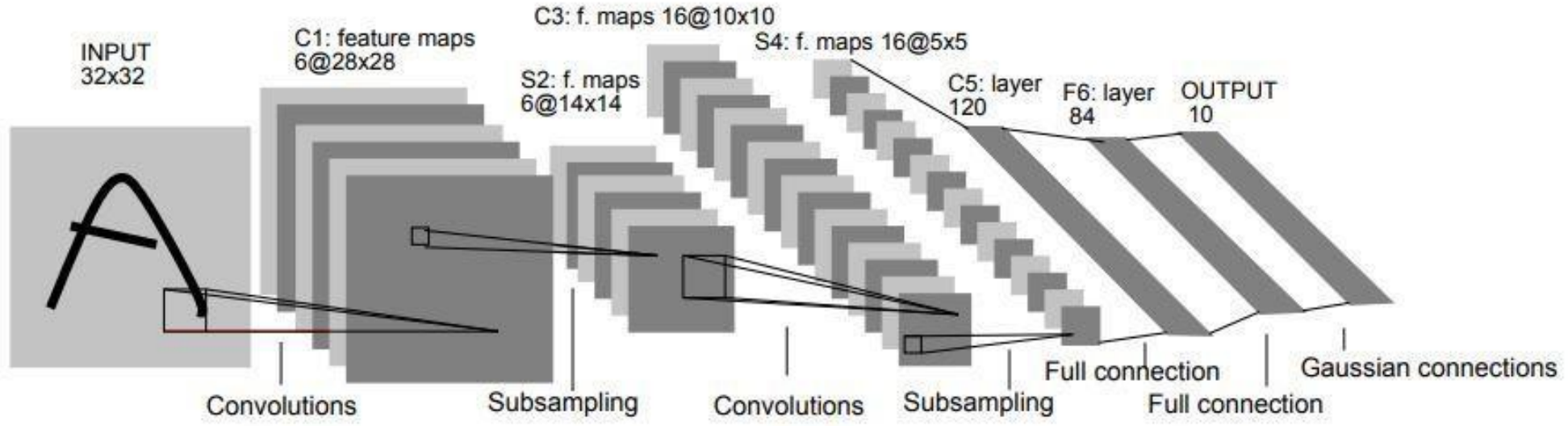
- In 1998, the LeNet-5 architecture was introduced in a research paper titled “Gradient-Based Learning Applied to Document Recognition” by Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner.

- It is one of the earliest and most basic CNN architecture.

- **Architecture:**
  - It consists of **7 layers**.
  - The first layer consists of an input image with dimensions of **32×32**.
  - It is convolved with 6 filters of size 5×5 resulting in dimension of **28x28x6**.
  - The second layer is a Pooling operation which filter size 2×2 and stride of 2. Hence the resulting image dimension will be **14x14x6**.
  - The third layer involves in a convolution operation with 16 filters of size **5×5**.
  - The fourth layer is a pooling layer with similar filter size of 2×2 and stride of 2. Thus, the resulting image dimension will be reduced to **5x5x16**.

LeNet-5 CNN Architecture

- The fifth layer is a fully connected convolutional layer with **120** filters each of **size 5×5**. In this layer, each of the 120 units in this layer will be connected to **the 400 (5x5x16)** units from the previous layers.
- The sixth layer is also a fully connected layer with **84** units.
- The final seventh layer will be a **softmax** output layer with ‘n’ possible classes depending upon the number of classes in the dataset.



AlexNet CNN Architecture

- AlexNet is one of the most popular neural network architectures to date. It was proposed by Alex Krizhevsky for the ImageNet Large Scale Visual Recognition Challenge (ILSVRV), and is based on convolutional neural networks.

- The challenge was to develop a Deep Convolutional Neural Network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 dataset into more than 1000 different categories.

- **Architecture:**
  - It consists of 8 layers in total, out of which the first 5 are convolutional layers and the last 3 are fully-connected.
  - The first two convolutional layers are connected to overlapping max-pooling layers to extract a maximum number of features. The third, fourth, and fifth convolutional layers are directly connected to the fully-connected layers.

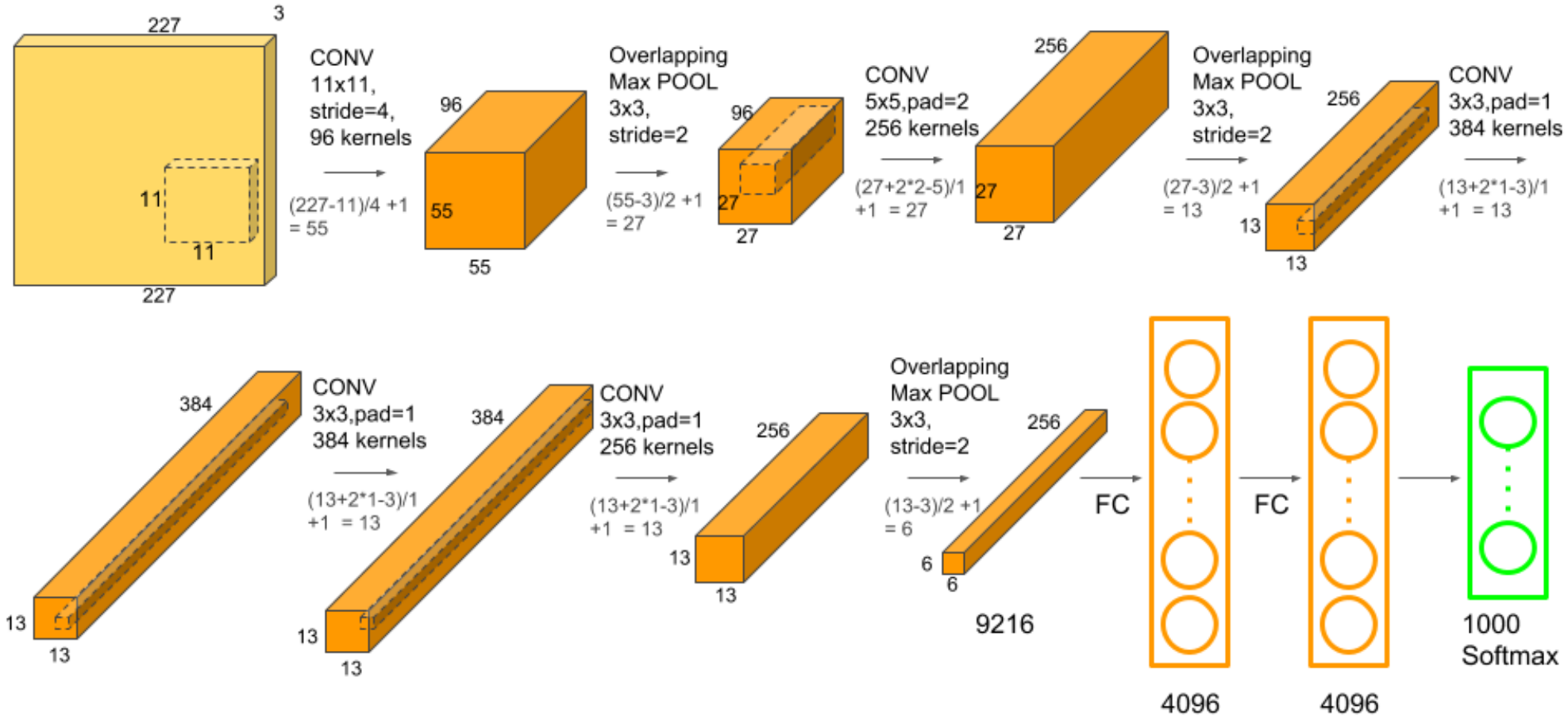
AlexNet CNN Architecture

- All the outputs of the convolutional and fully-connected layers are connected to ReLu non-linear activation function.
- The final output layer is connected to a softmax activation layer, which produces a distribution of 1000 class labels.

- The input dimensions of the network are (256 × 256 × 3), meaning that the input to AlexNet is an RGB (3 channels) image of (256 × 256) pixels.

- There are more than 60 million parameters and 650,000 neurons involved in the architecture.
- To reduce overfitting during the training process, the network uses dropout layers. The neurons that are “dropped out” do not contribute to the forward pass and do not participate in backpropagation. These layers are present in the first two fully-connected layers.

AlexNet CNN Architecture



Other CNN Architecture

- VGG16
- VGG19
- RasNet-18
- ResNet-50
- GoogleNet etc.

Challenges of Deep Learning

- Works better with large amount of data •
- Some models are very hard to train, may take weeks or months
- Overfitting
- Black box and hence may have regulatory challenges