Machine Learning Lecture Series

UNIT: 02

Lecture: 04

Machine Learning

CSE21816

Course Instructor

Dr. Tamal Ghosh

Associate Professor

Computer Science and Engineering, Adamas University

tamal.ghosh1@adamasuniversity.ac.in

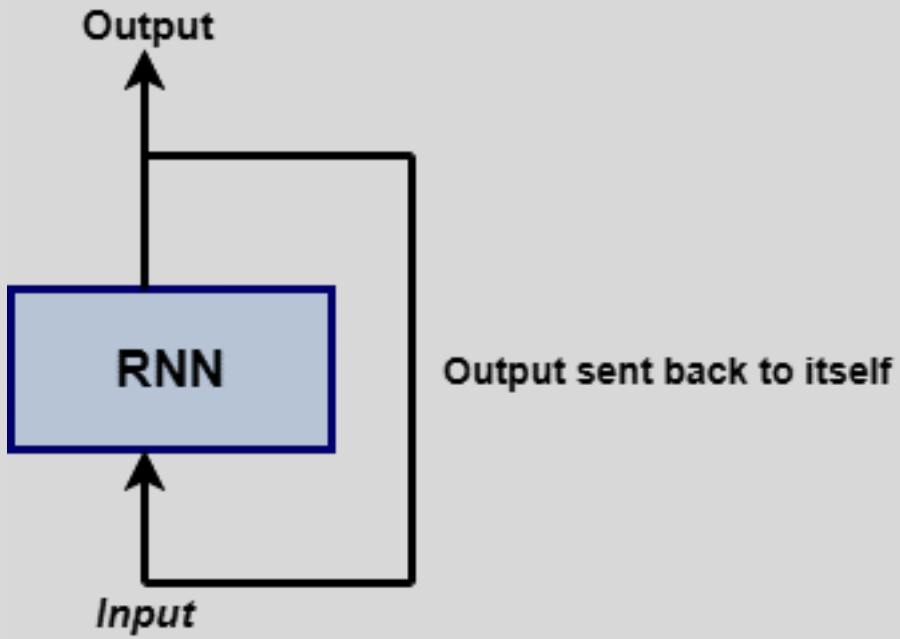
Recurrent Neural Network (RNN)

What is RNN?

- A recurrent neural network (RNN) is a kind of artificial neural network mainly used in speech recognition and natural language processing (NLP). RNN is used in deep learning and in the development of models that imitate the activity of neurons in the human brain.
- Recurrent Networks are designed to recognize patterns in sequences of data, such as text, genomes, handwriting, the spoken word, and numerical time series data emanating from sensors, stock markets, and government agencies.
- A recurrent neural network looks similar to a traditional neural network except that a memory-state is added to the neurons. The computation is to include a simple memory.

RNN

- The recurrent neural network is a type of deep learning algorithm, which follows a sequential approach. In RNN, we always assume that each input and output is dependent on all other layers. It is called recurrent because they sequentially perform mathematical computations.



Application of RNN

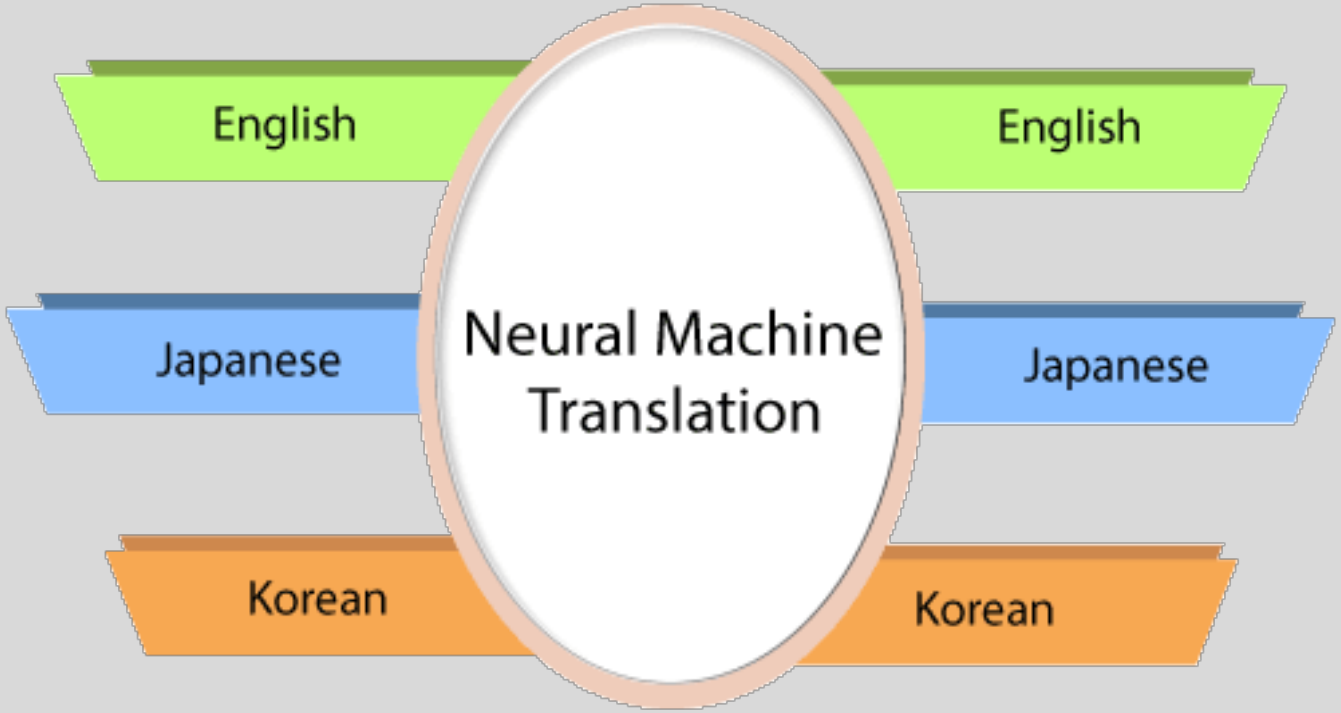
- RNN has multiple uses when it comes to predicting the future. In the financial industry, RNN can help predict stock prices or the sign of the stock market direction (i.e., positive or negative).

- RNN is used for an autonomous car as it can avoid a car accident by anticipating the route of the vehicle.

- RNN is widely used in image captioning, text analysis, machine translation, and sentiment analysis. For example, one should use a movie review to understanding the feeling the spectator perceived after watching the movie. Automating this task is very useful when the movie company can not have more time to review, consolidate, label, and analyze the reviews. The machine can do the job with a higher level of accuracy.

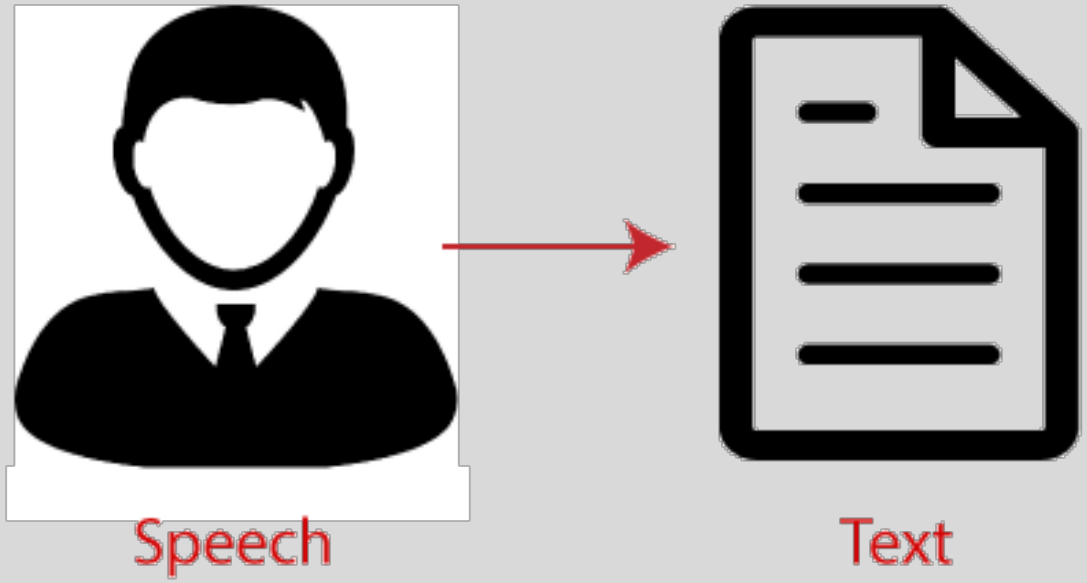
Machine Translation

- We make use of Recurrent Neural Networks in the translation engines to translate the text from one to another language. They do this with the combination of other models like LSTM (Long short-term memory)s.



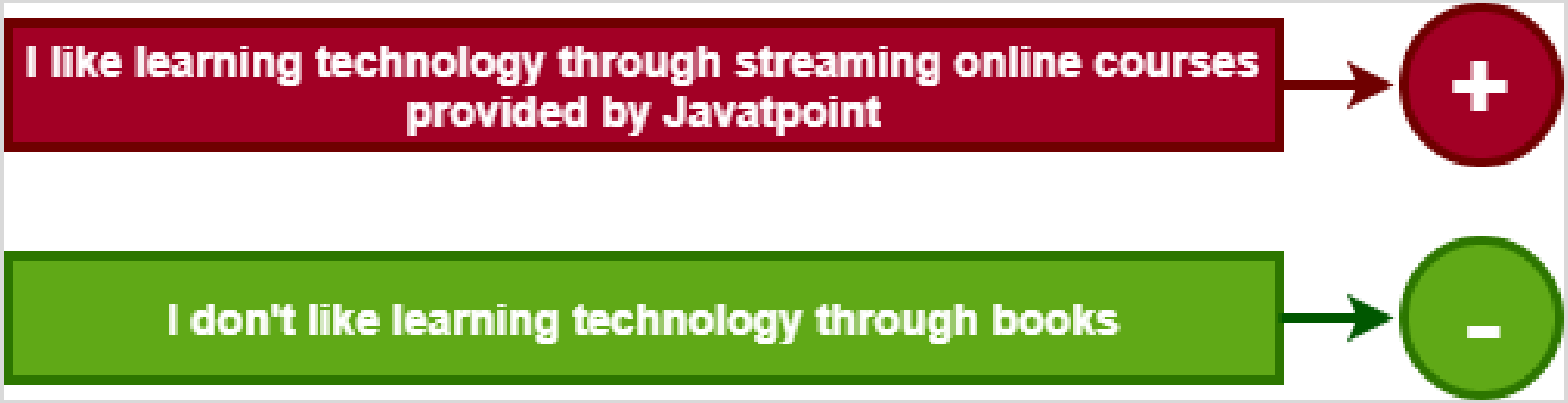
Speech Recognition

- Recurrent Neural Networks has replaced the traditional speech recognition models that made use of Hidden Markov Models. These Recurrent Neural Networks, along with LSTMs, are better poised at classifying speeches and converting them into text without loss of context.



Sentiment Analysis

- We make use of sentiment analysis to positivity, negativity, or the neutrality of the sentence. Therefore, RNNs are most adept at handling data sequentially to find sentiments of the sentence.



Automatic Image Tagger

- RNNs, in conjunction with convolutional neural networks, can detect the images and provide their descriptions in the form of tags. For example, a picture of a fox jumping over the fence is better explained appropriately using RNNs.

A White Dog Jumping into the water

Limitations of RNN

- RNN is supposed to carry the information in time. However, it is quite challenging to propagate all this information when the time step is too long. When a network has too many deep layers, it becomes untrainable. This problem is called: vanishing gradient problem.

- If we remember, the neural network updates the weight use of the gradient descent algorithm. The gradient grows smaller when the network progress down to lower layers.

- The gradient stays constant, meaning there is no space for improvement. The model learns from a change in its gradient; this change affects the network's output. If the difference in the gradient is too small (i.e., the weight change a little), the system can't learn anything and so the output. Therefore, a system facing a vanishing gradient problem cannot converge towards the right solution.

Steps in RNN

- The recurrent network first performs the conversion of independent activations into dependent ones. It also assigns the same weight and bias to all the layers, which reduces the complexity of RNN of parameters. And it provides a standard platform for memorization of the previous outputs by providing previous output as an input to the next layer.
- These three layers having the same weights and bias, combine into a single recurrent unit.
- For calculating the current state-

$\mathbf{h_t = f(h_{t-1}, X_t)}$ Where h_t= **current state,** H_{t-1}= **previous state,** X_t= **input state**

- To apply the activation function (sigma: tanh or sigmoid) tanh, we have-

$\mathbf{h_t = \tanh (W_h h_t\text{-}1 + W_x X_t)}$ Where: W_h = **weight of recurrent neuron and,** W_x = **weight of the input neuron**

The formula for calculating output:

$$\mathbf{Y_t = W_h h_t}$$

Training through RNN

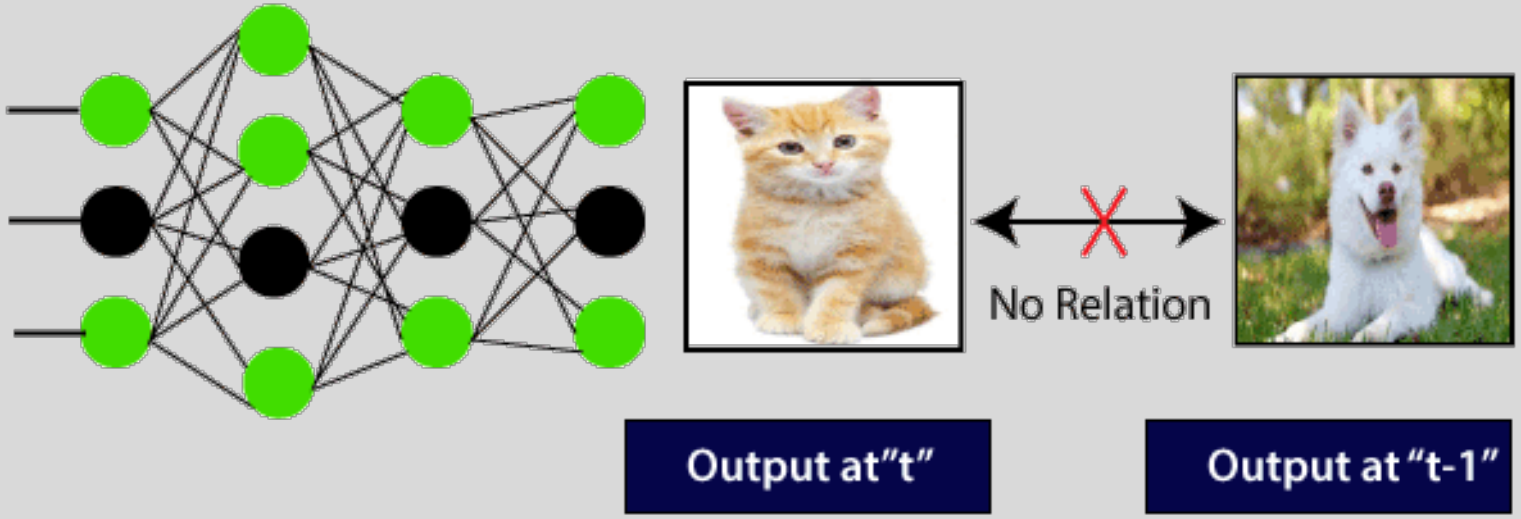
- The network takes a single time-step of the input.
- We can calculate the current state through the current input and the previous state.
- Now, the current state through ht-1 for the next state.
- There is n number of steps, and in the end, all the information can be joined.
- After completion of all the steps, the final step is for calculating the output.
- At last, we compute the error by calculating the difference between actual output and the predicted output.
- The error is backpropagated to the network to adjust the weights and produce a better outcome.

Example Problem

Consider an image classification use-case where we have trained the neural network to classify images of some animals.

So, let's feed an image of a cat or a dog; the network provides an output with the corresponding label to the picture of a cat or a dog.

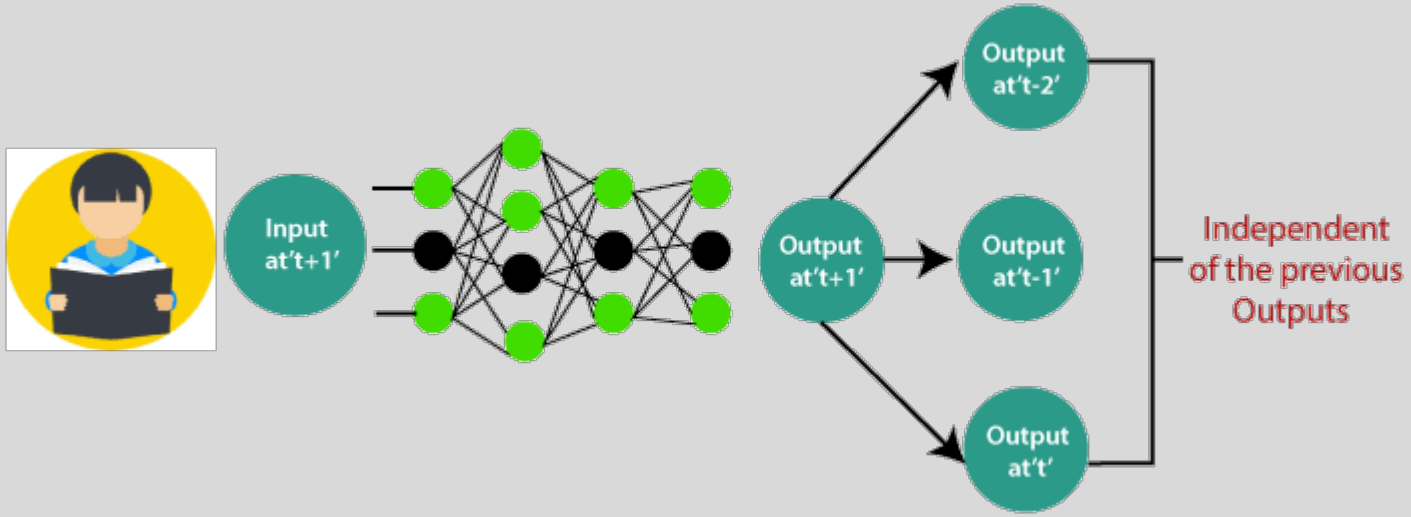
See the below diagram:



Contd...

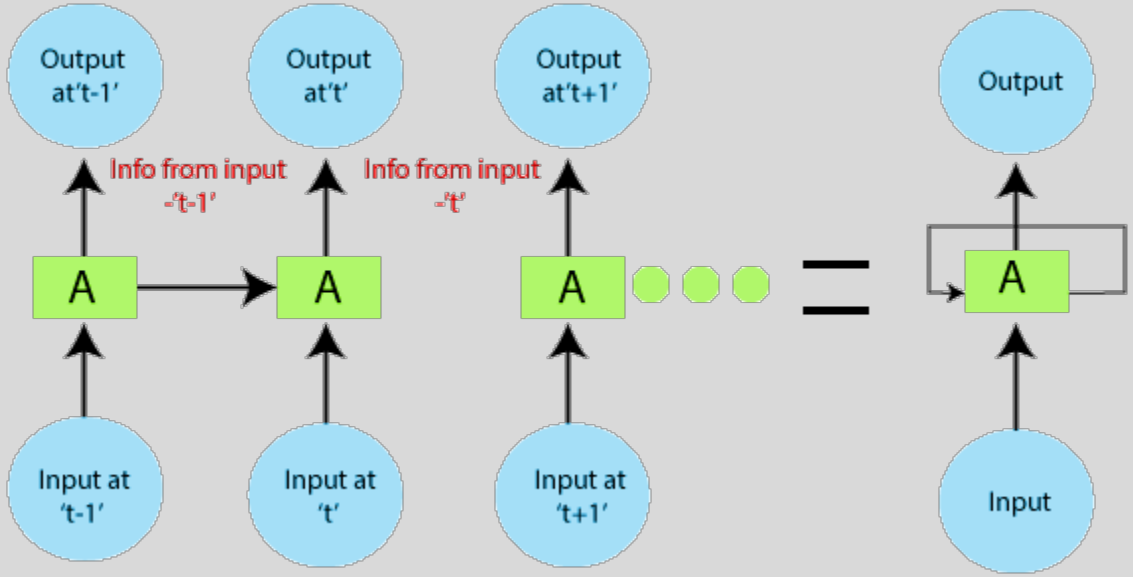
Here, the first output being a cat will not influence the previous output, which is a dog. This means that output at a time 't' is autonomous of output at the time 't-1?.

Consider the scenario where we will require the use of the last obtained output:



Contd...

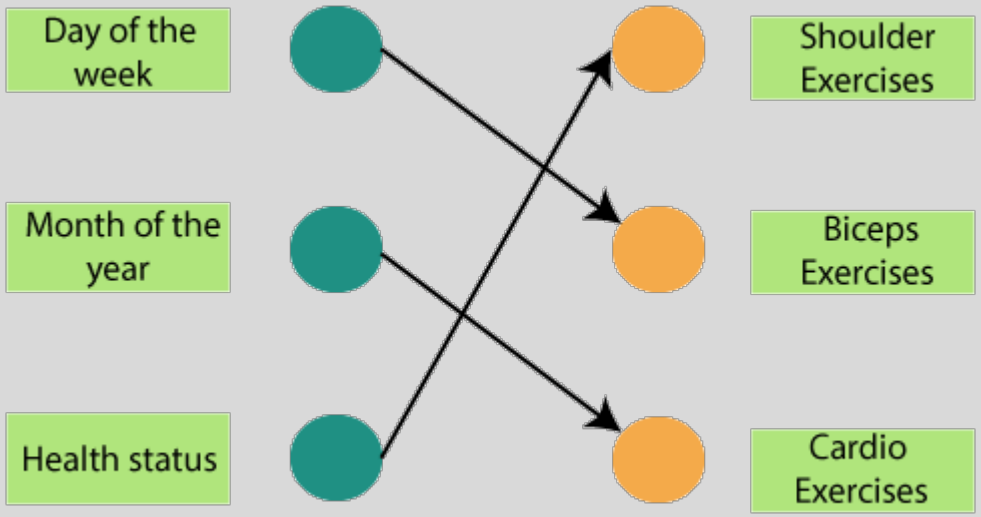
- The concept is the same as reading a book. With every page we move forward into, we need the understanding of previous pages to make complete sense of the information in most of the cases.
- With the help of the feed-forward network, the new output at the time 't+1? has no relation with outputs at either time t, t-1, t-2.
- So, the feed-forward network cannot be used when predicting a word in a sentence as it will have no absolute relation with the previous set of words. But, with the help of RNN, this challenge can be overcome.

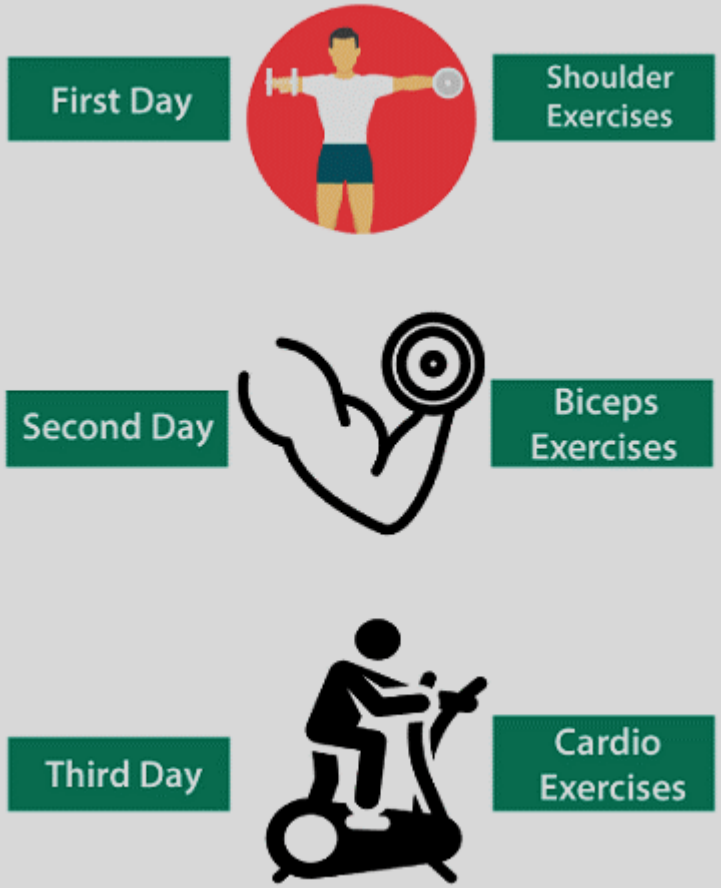


Gym Exercise Example

- We go to the gym regularly, and the trainer has given us the schedule of our workout: →

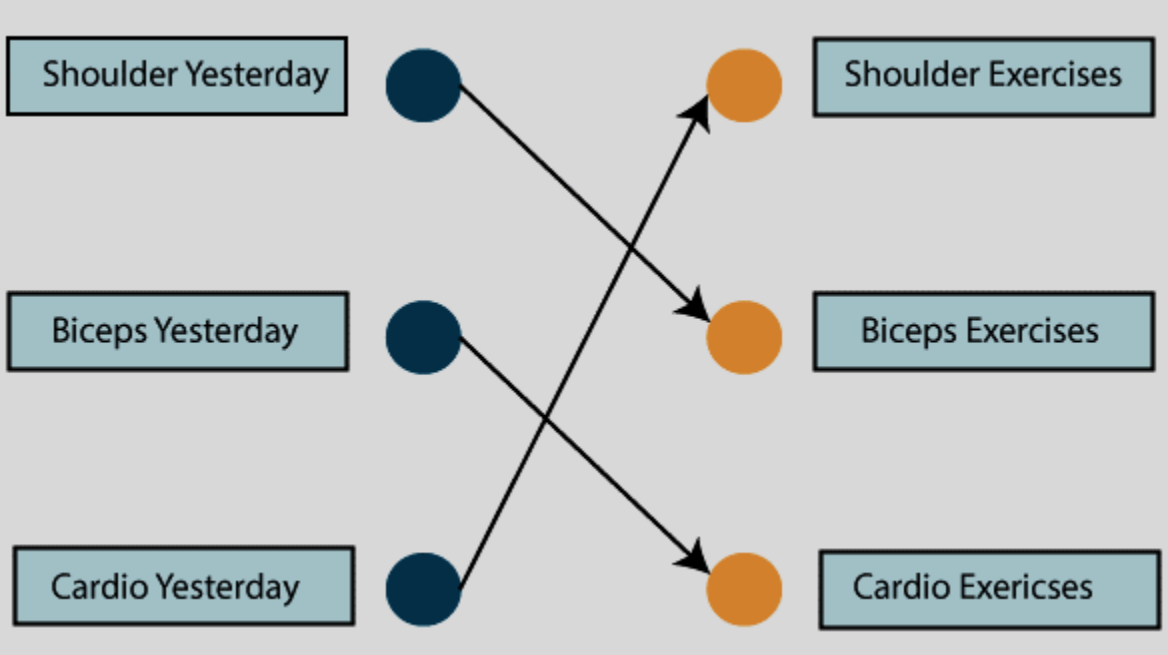
- Note that all the exercises are repeated in a proper order every week. Let us use a feed-forward network to trying and predicting the types of exercises.





Contd...

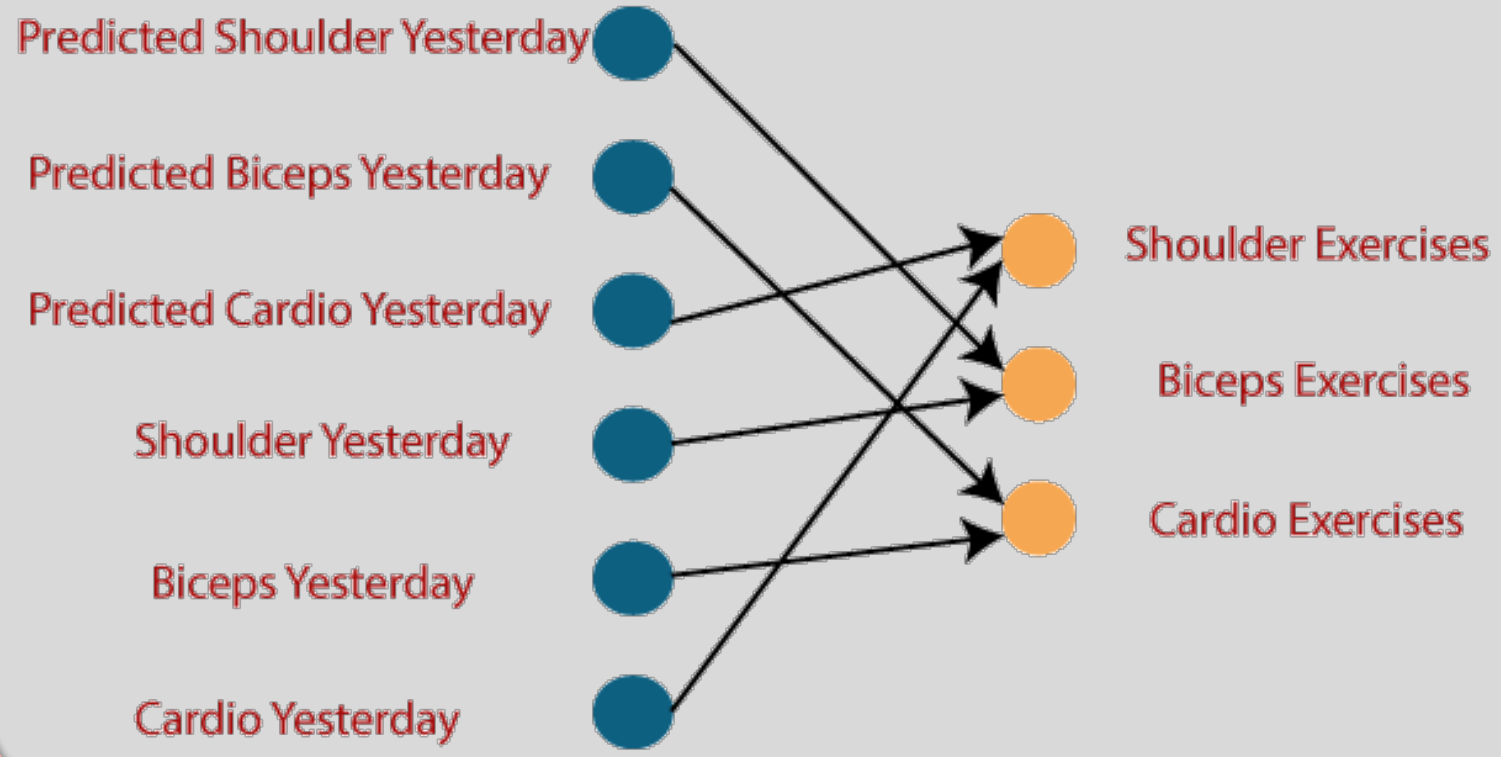
- The inputs are the **day, month,** and **health status**. A neural network has been trained using these inputs to provide the prediction of the exercise.
- However, this will not very accurate, considering the input. To fix this, we make use of the concept of Recurrent Neural Networks, as shown:



In this case, find the inputs to be workout done on the previous day. So if we did a shoulder exercise yesterday, we could do a bicep exercise today, and this goes on for the rest of the week.

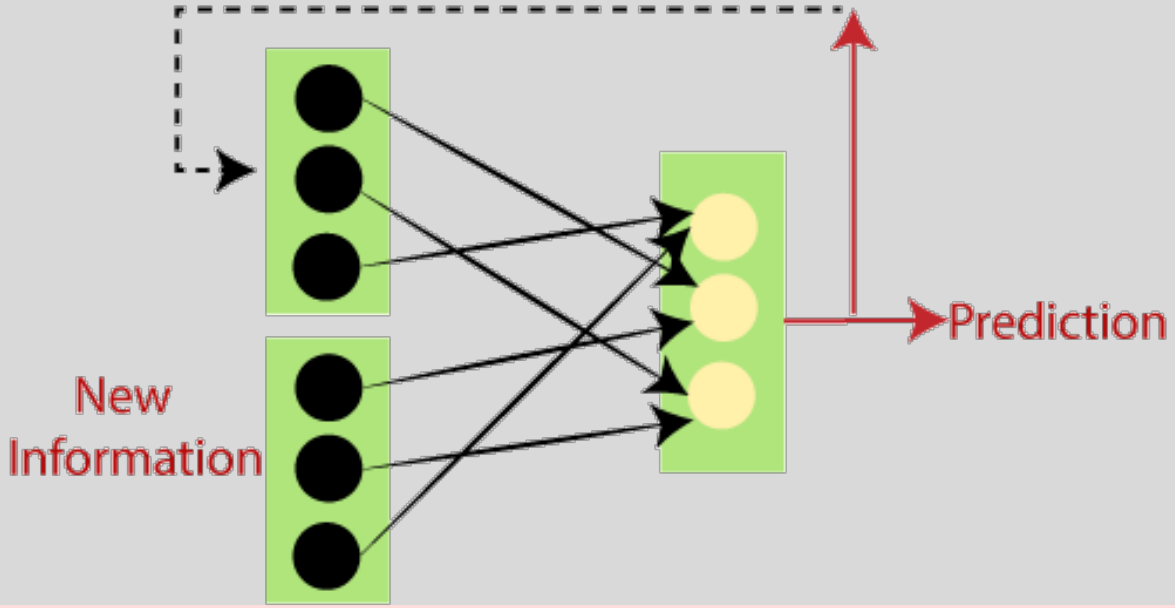
Contd...

- However, if we happen to miss a day at the gym, the data from the previously attended timestamp can be considered below.



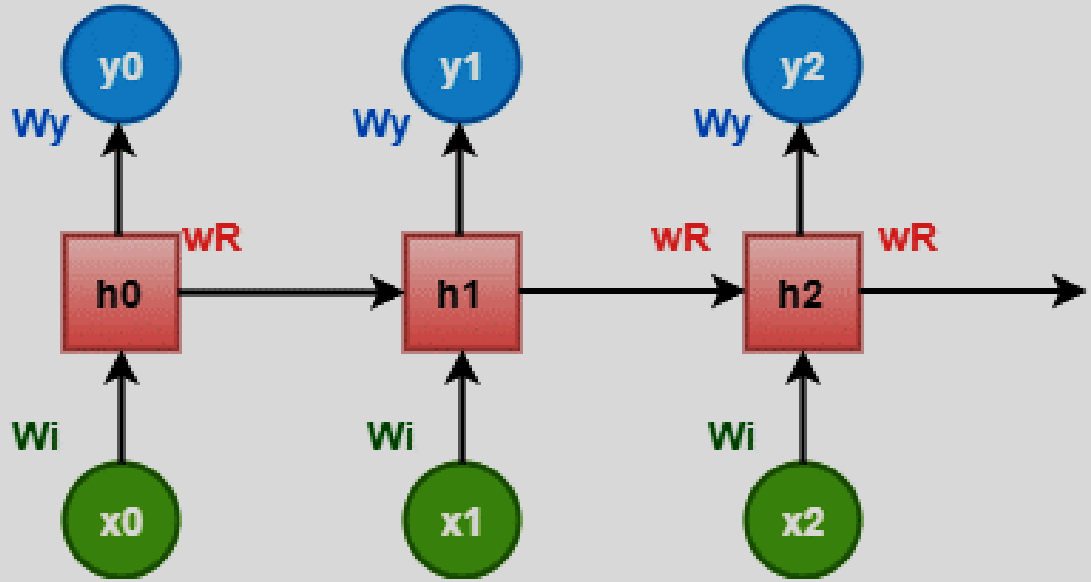
Contd...

- If a model is experienced based on the data it obtains from the last exercises, the output from the model will be accurate.
- To sum it, let us convert the data into vectors.
- Where vectors are numbers which are input to the **model** to **denote** if we have done the exercise or not.



Cost Function for Linear Regression

- So, if we have a shoulder exercise, the corresponding node will be '1', and the rest of the exercise nodes will be mapped to '0'.
- We have to check the math behind the working of the neural network.



$$h^{(t)} = g_h\left(w_i x^{(t)} + w_R h^{(t-1)} + b_h\right)$$

$$y^{(t)} = g_y\left(w_y h^{(t)} + b_y\right)$$

Contd…

- examine 'w' to be the weight matrix and 'b' as the bias:
- At time t=0, the input is 'x0, and the task is to figure out what is 'h0'. Substituting t=0 in the equation and acquiring the function h(t) value. The next value of 'y0' is finding out using the previously calculated values when applied to the new formula.
- The same process is repeated through all of the timestamps in the model to train a model.

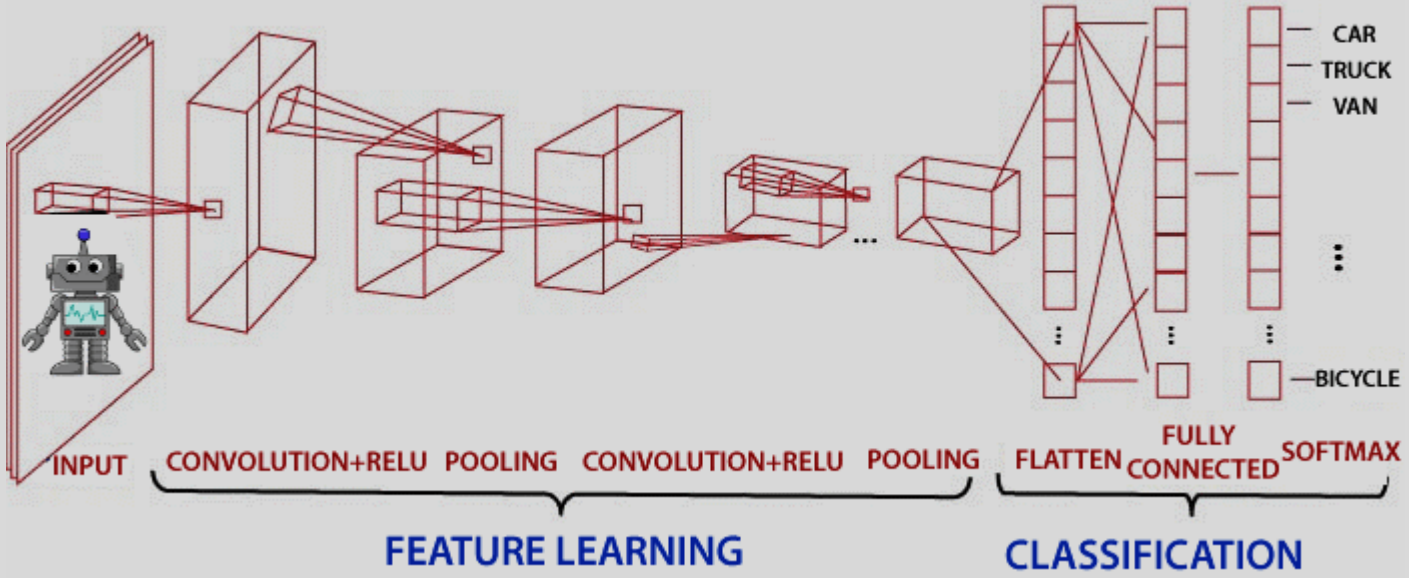
Training of RNN

- Recurrent Neural Networks use a **backpropagation algorithm** for training, but it is applied for each timestamp. It is commonly known as **Back-propagation** by Time (BTT).
- Some issues with Back-propagation, such as:
  - Vanishing Gradient
  - Exploding Gradient
- Details are in the notes.

Convolutional Neural Network (CNN)

CNN

- CNN is one of the techniques to do image classification and image recognition in neural networks.
- It is designed to process the data by multiple layers of arrays. This type of neural network is used in applications like image recognition or face recognition.
- The primary difference between CNN and other neural network is that CNN takes input as a two-dimensional array. And it operates directly on the images rather than focusing on feature extraction which other neural networks do.

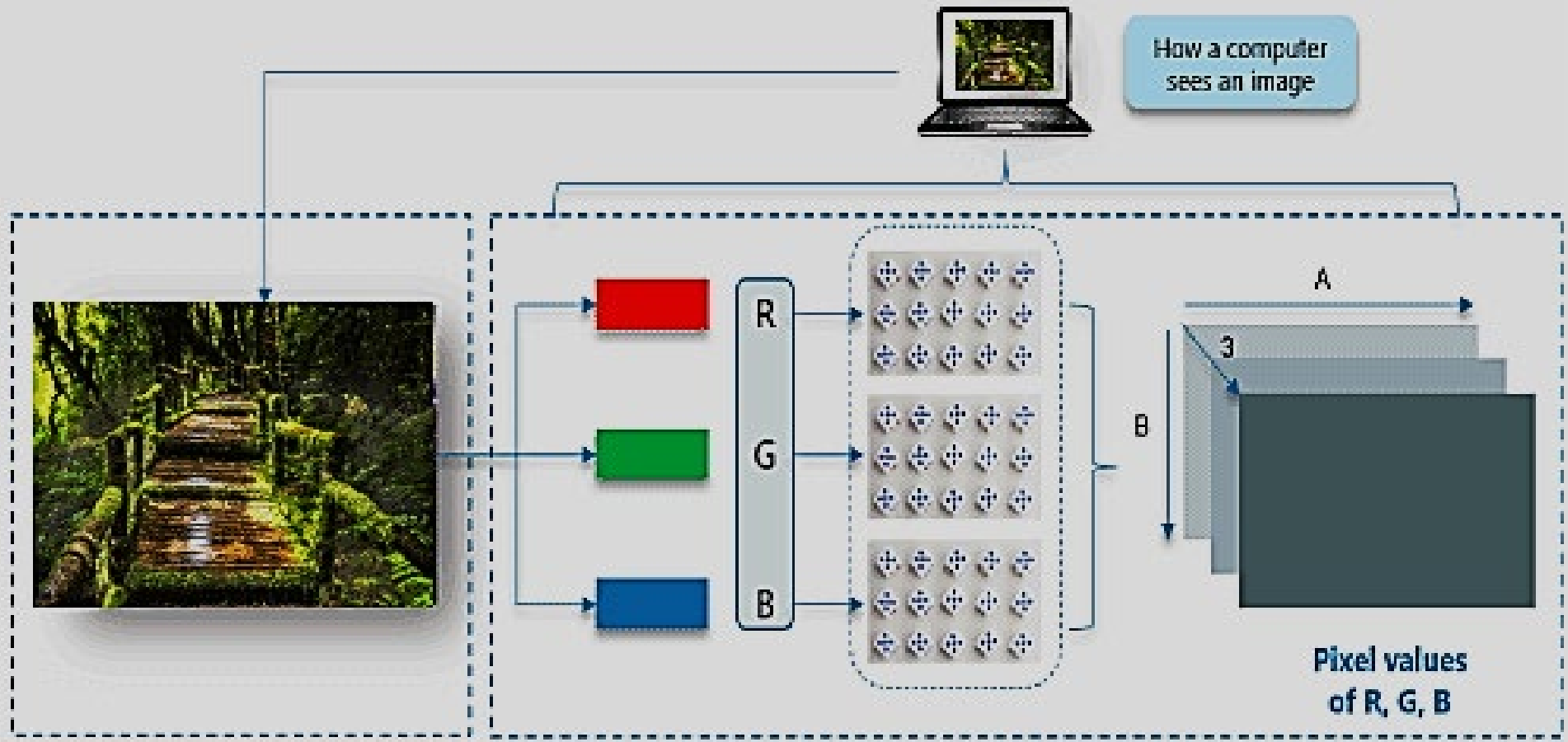


CNN

- The dominant approach of CNN includes the solution for problems of recognition. Some companies like **Google** and **Facebook** have invested in the field research and development concerning recognition projects to get activities done with higher speed.
- Scene labeling, object detection, and face recognition, etc. are some of the areas where Convolutional Neural Network works.
- Convolutional Neural Network (CNN or ConvNet) is a type of feed-forward artificial network where the connectivity pattern between its neurons is inspired by the organization of the animal **visual cortex**.

How Does a Computer read an image?

- The image is broken into 3 color channels which are Red, Green, and Blue. Each of these color channels is mapped to the image's pixel.



Contd…

- Some neurons fire when exposed to vertical edges and some when exposed to horizontal or diagonal edges.
- CNN utilizes spatial correlations which exist with the input data. Each concurrent layer of the neural network connects some input neurons.
- This region is called a **local receptive field**. The local receptive field focuses on hidden neurons.
- The hidden neuron processes the input data inside the mentioned field, not realizing the changes outside the specific boundary.

CNN has the following 4 layers

- Convolutional Layer
- ReLU Layer
- Pooling Layer
- Fully Connected Layer

Convolutional layer

- Convolution layer is the first layer to derive features from the input image. The convolutional layer conserves the relationship between pixels by learning image features using a small square of input data. It is the mathematical operation which takes two inputs such as **image matrix** and **kernel** or **any filter**.
- The dimension of image matrix is **h×w×d**.
- The dimension of any filter is **f_h×f_w×d**.
- The dimension of output is **(h-f_h+1)×(w-f_w+1)×1**.

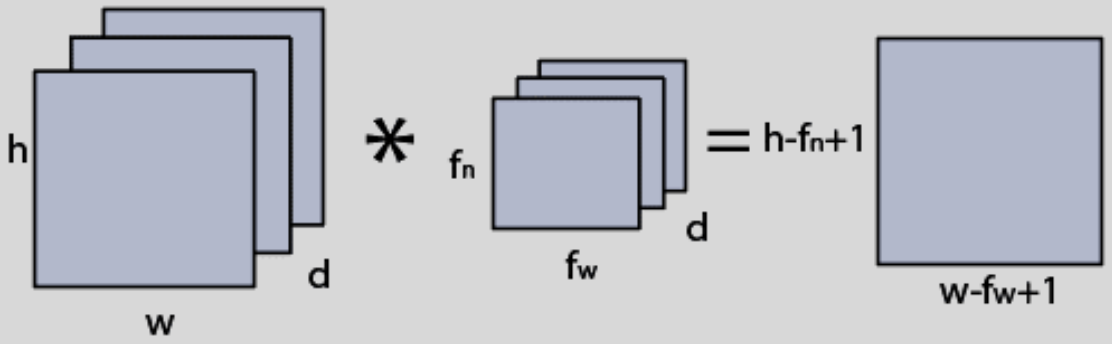


Image matrix multiplies kernl or filter matrix

Contd...

- Let's start with consideration a 5*5 image whose pixel values are 0, 1, and filter matrix 3*3 as:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$5 \times 5 -$ Image Matrix $\quad 3 \times 3 -$ Filter Matrix

- The convolution of 5*5 image matrix multiplies with 3*3 filter matrix is called "**Features Map**" and show as an output.

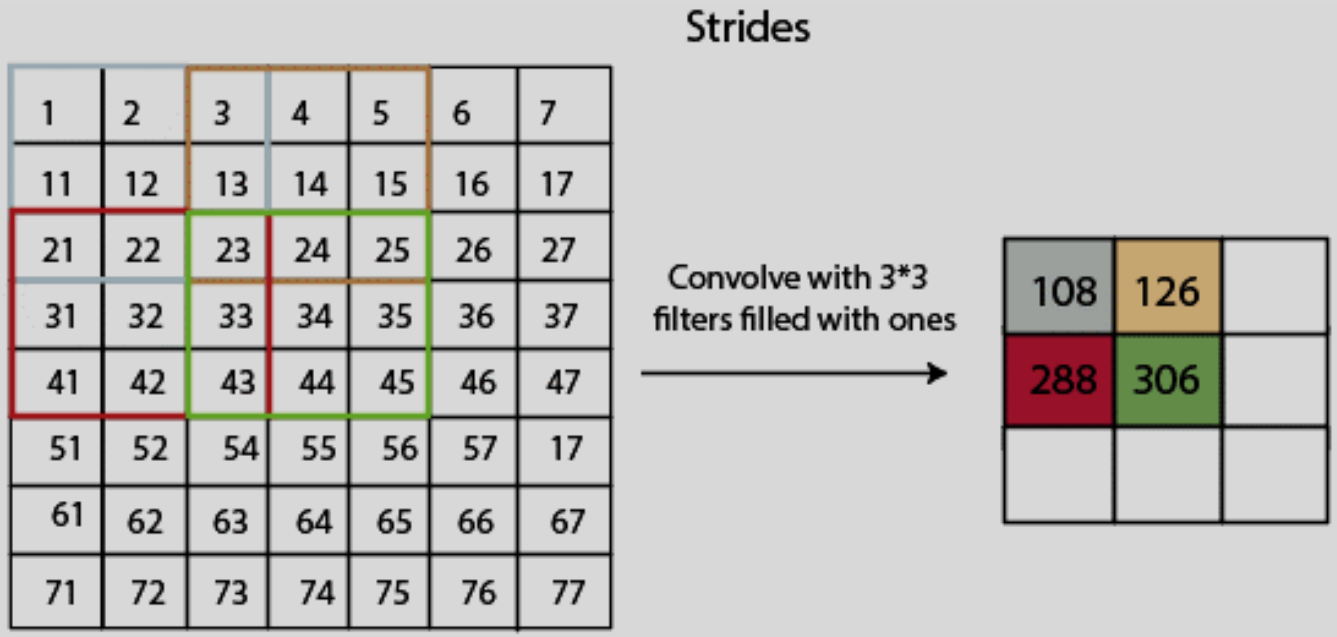
$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 3 & 4 \\ 2 & 4 & 3 \\ 2 & 3 & 4 \end{bmatrix}$$

Convolved Feature

- Convolution of an image with different filters can perform an operation such as blur, sharpen, and edge detection by applying filters.

Stride

- Stride is the number of pixels that shift over the input matrix. When the stride = 1, then we move the filters to 1 pixel at a time and similarly, if the stride = 2, then we move the filters to 2 pixels at a time.



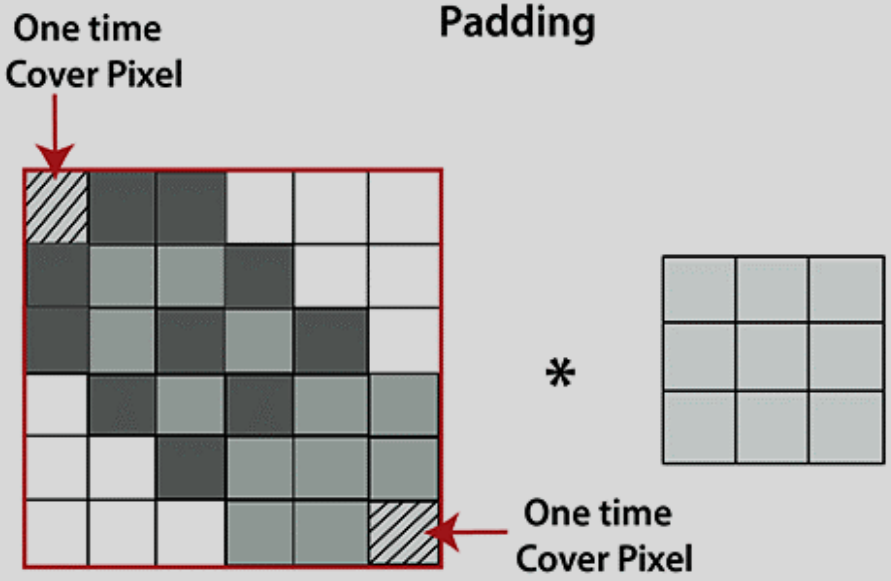
Padding

- Padding plays a crucial role in building the convolutional neural network. If the image shrinks and if we take a neural network with 100's of layers on it, it will give us a small image after filtering.
- If we take a three-by-three filter on top of a grayscale image and do the convolution then what will happen?

It is clear from the picture that the pixel in the corner will only get covers one time, but the middle pixel will get covered more than once. It means that we have more information on that middle pixel, so there are two downsides:

- Shrinking outputs
- Losing information on the corner of the image.

To overcome this, padding is introduced.
"Padding is an additional layer which can add to the border of an image."



Contd...

- Let's start with consideration a 5*5 image whose pixel values are 0, 1, and filter matrix 3*3 as:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

5×5 – Image Matrix 3×3 – Filter Matrix

- The convolution of 5*5 image matrix multiplies with 3*3 filter matrix is called "**Features Map**" and show as an output.

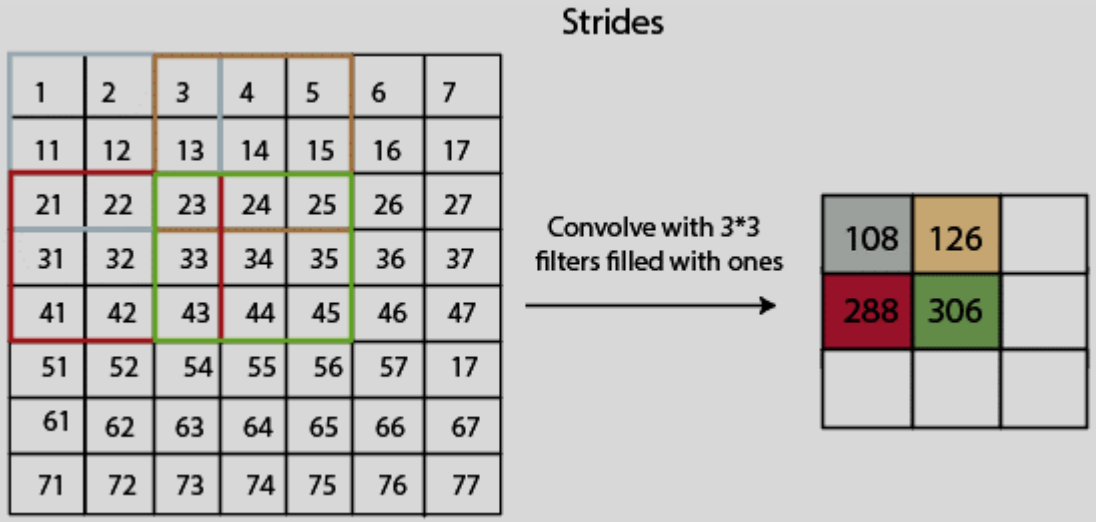
$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 3 & 4 \\ 2 & 4 & 3 \\ 2 & 3 & 4 \end{bmatrix}$$

Convolved Feature

- Convolution of an image with different filters can perform an operation such as blur, sharpen, and edge detection by applying filters.

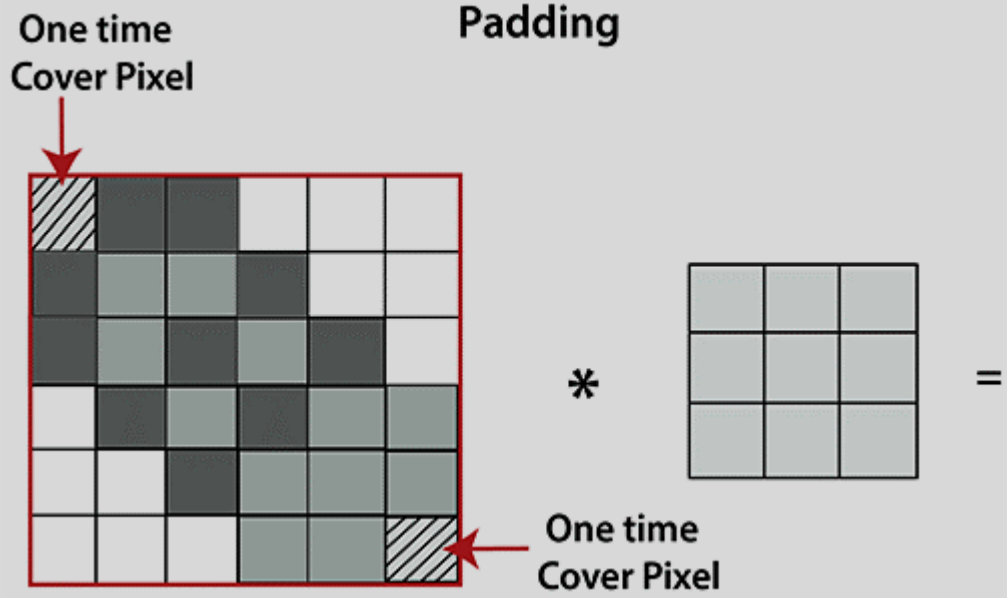
Strides

- Stride is the number of pixels which are shift over the input matrix. When the stride is equaled to 1, then we move the filters to 1 pixel at a time and similarly, if the stride is equaled to 2, then we move the filters to 2 pixels at a time. The following figure shows that the convolution would work with a stride of 2.



Padding

- Padding plays a crucial role in building the convolutional neural network. If the image shrinks and if we take a neural network with 100's of layers on it, it will give us a small image after filtering.
- If we take a three-by-three filter on top of a grayscale image and do the convolving then what will happen?



The picture shows that the pixel in the corner will only get covered one time, but the middle pixel will get covered more than once. It means that we have more information on that middle pixel, so there are two downsides:

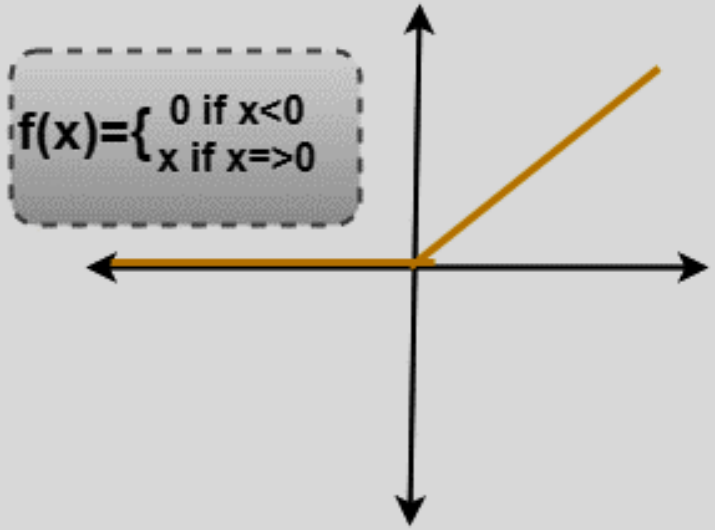
- Shrinking outputs
- Losing information on the corner of the image

To overcome this, we have introduced padding to an image.

"Padding is an additional layer which can add to the border of an image."

ReLU Layer

- **Rectified Linear unit(ReLU)** transform functions only activates a node if the input is above a certain quantity. While the data is below zero, the output is zero, but when the input rises above a certain threshold. It has a linear relationship with the dependent variable.
- In this layer, we remove every negative value from the filtered images and replaces them with zeros.
- It is happening to avoid the values from adding up to zero.
-

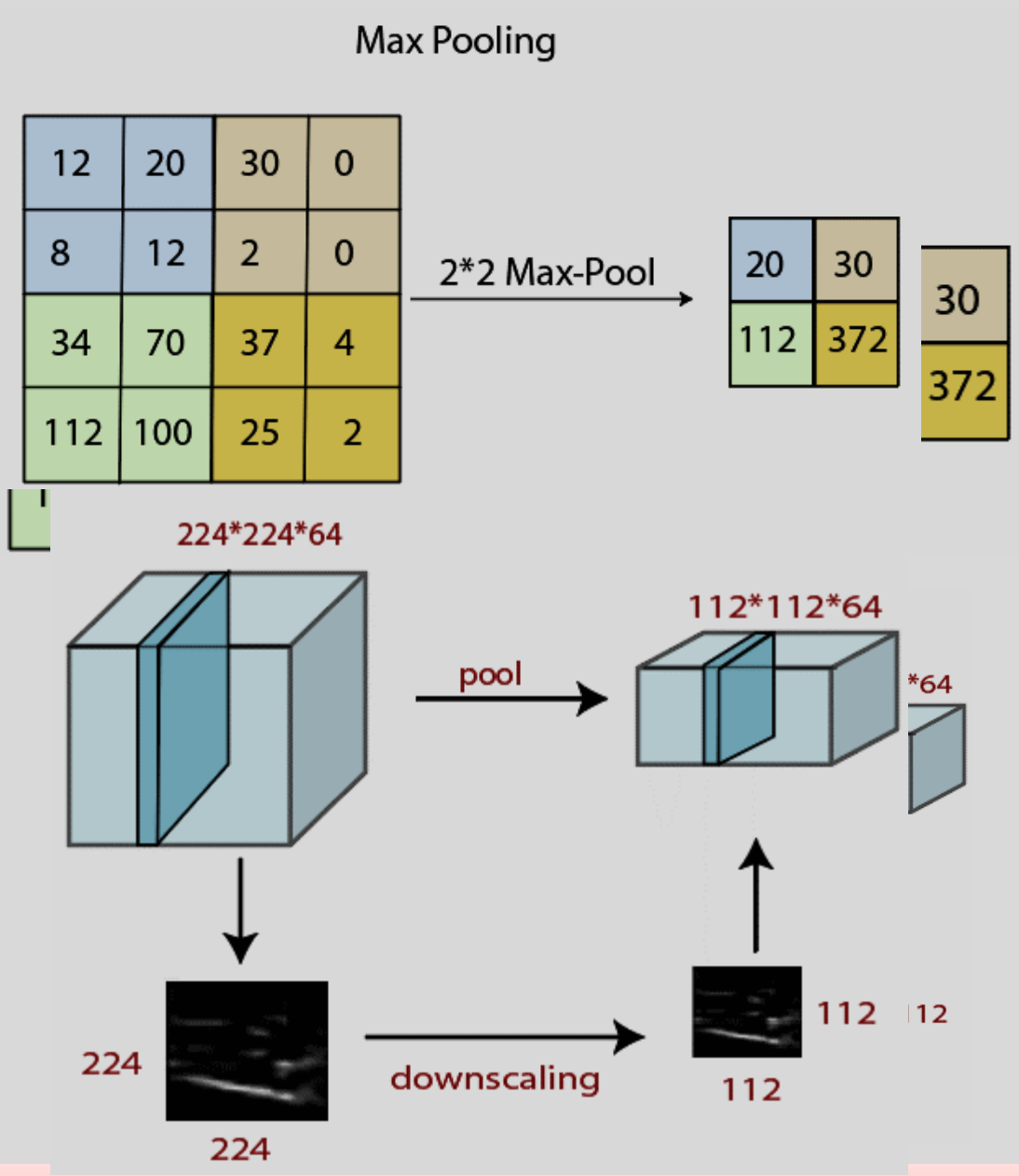


Pooling Layer

- Pooling layer plays an important role in the pre-processing of an image. The pooling layer reduces the number of parameters when the images are too large. Pooling is "downscaling" of the image obtained from the previous layers. It can be compared to shrinking an image to reduce its pixel density.
- Spatial pooling is also called downsampling or subsampling, which reduces the dimensionality of each map but retains the important information.
- There are the following types of spatial pooling:
  - MAX Pooling
  - Average Pooling

Max Pooling

- Max pooling is a sample-based discretization process. Its main objective is to downscale an input representation, reducing its dimensionality and allowing for the assumption to be made about features contained in the sub-region discarded.
- Max pooling is done by applying a max filter to non-overlapping sub-regions of the initial representation.



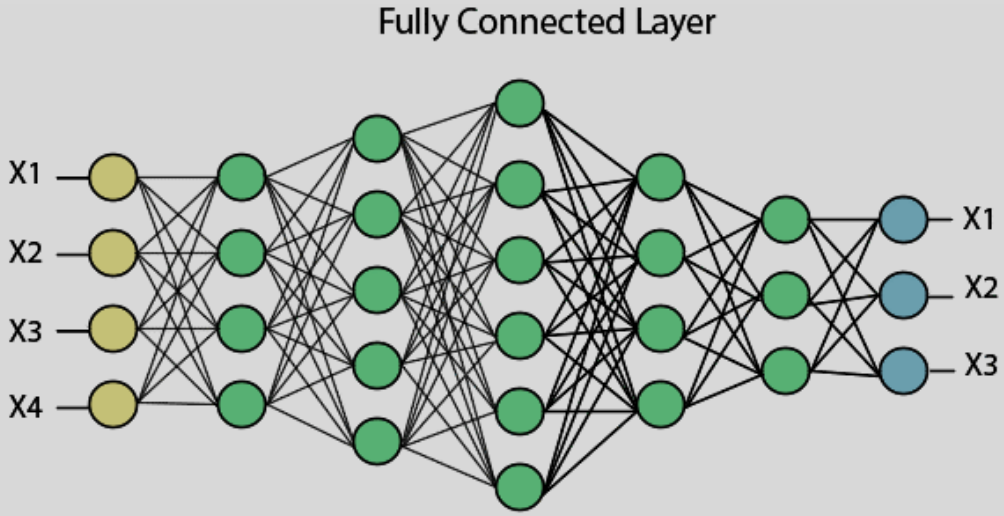
Average Pooling

- Down-scaling will perform through average pooling by dividing the input into rectangular pooling regions and computing the average values of each region.

- **Sum Pooling**
- The sub-region for **sum pooling** or **mean pooling** are set exactly the same as for **max-pooling** but instead of using the max function we use sum or mean.

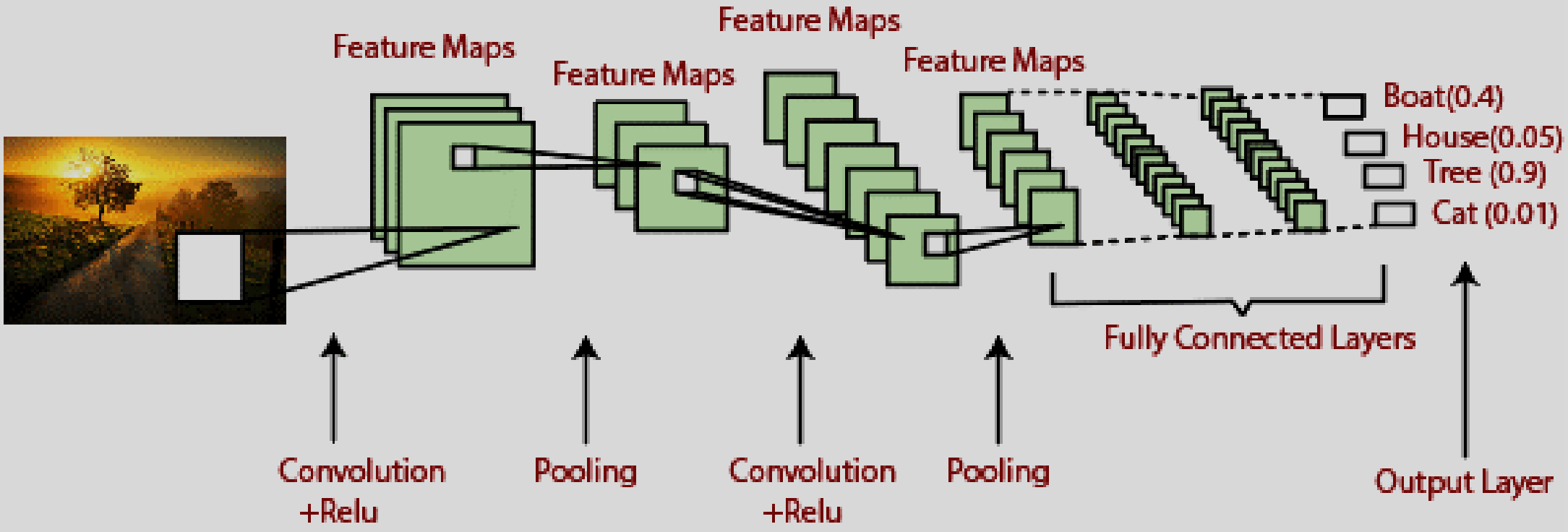
Fully Connected Layer

- The fully connected layer is a layer in which the input from the other layers will be flattened into a vector and sent. It will transform the output into the desired number of classes by the network.



- In the above diagram, the feature map matrix will be converted into the vector such as **x1, x2, x3... xn** with the help of fully connected layers. We will combine features to create a model and apply the activation function such as **softmax** or **sigmoid** to classify the outputs as a car, dog, truck, etc.

Contd...


Feature Maps
Feature Maps
Feature Maps
Feature Maps
Boat(0.4)
House(0.05)
Tree (0.9)
Cat (0.01)
Fully Connected Layers
Convolution
+Relu
Pooling
Convolution
+Relu
Pooling
Output Layer