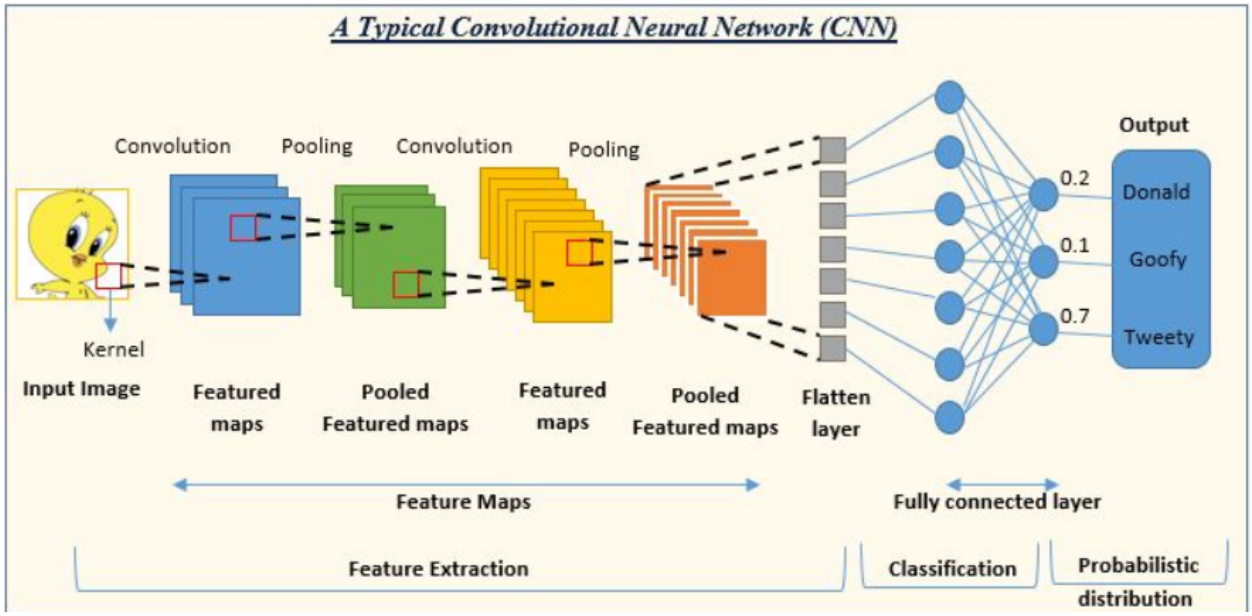
Report

Shounak Das
(21D070068)

Convolutional Neural Networks (CNNs):



A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics. The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted

region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.
Convolutional Neural Networks are very similar to ordinary Neural Networks : They are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still express a single differentiable score function: From the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer and all the tips/tricks we developed for learning regular Neural Networks still apply.

Architecture of CNN :

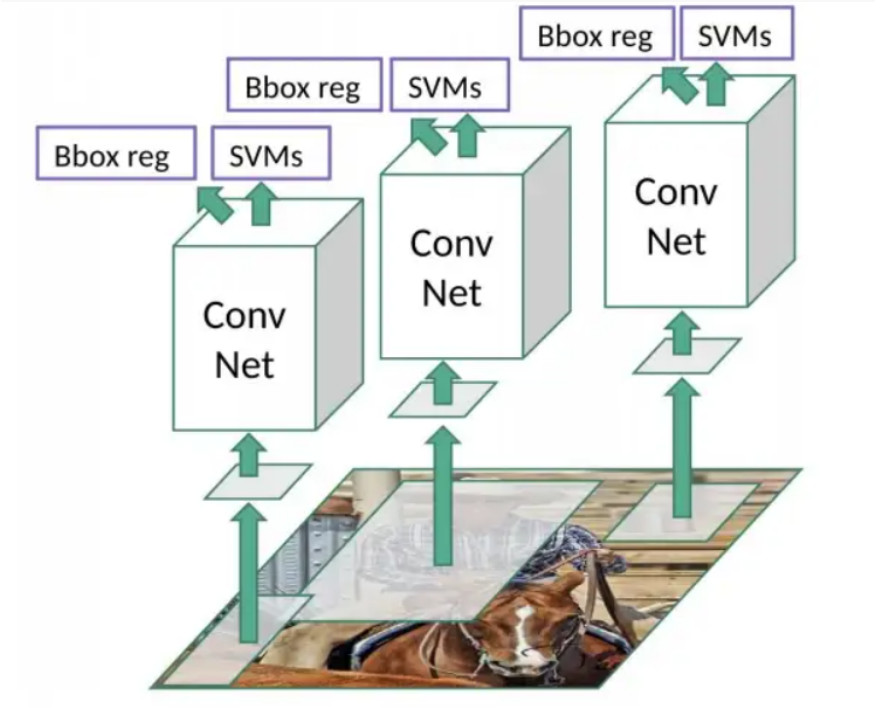
It has three layers namely, convolutional, pooling, and a fully connected layer. It is a class of neural networks and processes data having a grid-like topology. The convolution layer is the building block of CNN carrying the main responsibility for computation. Pooling reduces the spatial size of the representation and lessens the number of computations required. Whereas, the Fully Connected Layer is connected to both the layers, prior and the recent one.
To effectively communicate about the created models, it is imperative to use visual tools to communicate about the architecture of CNN. These tools help to create cnn diagrams by representing the model visually in an appealing manner. There are many tools that can be used to draw the architecture such as- Diagram.net NN- SVG Plot Neural Net TensorSpace.js Keras.js

Layers :

A ConvNet architecture is a list of Layers that transform the image volume into an output volume (e.g. holding the class scores) .
There are a few distinct types of Layers (e.g. CONV/FC/RELU/POOL are by far the most popular).

Each Layer accepts an input 3D volume and transforms it to an output 3D volume through a differentiable function.
Each Layer may or may not have parameters (e.g. CONV/FC do, RELU/POOL don't).
Each Layer may or may not have additional hyperparameters (e.g. CONV/FC/POOL do, RELU doesn't).

Region-based Convolutional Neural Networks (R-CNNs):

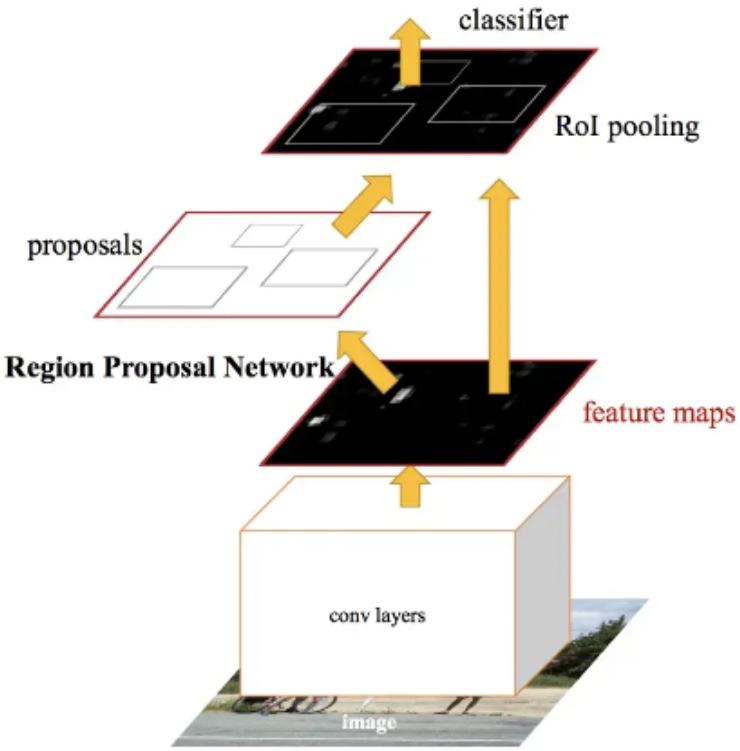


To bypass the problem of selecting a huge number of regions, we use selective search to extract just around 1000-2000 regions from the image and he called them region proposals. Therefore, now, instead of trying to classify a huge number of regions, you can just work with the 1000-2000 regions. These region proposals are generated using the selective search algorithm.

Problems with R-CNNs:

It still takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image. It cannot be implemented real time as it takes around 47 seconds for each test image. The selective search algorithm is a fixed algorithm. Therefore, no learning is happening at that stage. This could lead to the generation of bad candidate region proposals.

Fast R-CNN :



The approach is similar to the R-CNN algorithm. But, instead of feeding the region proposals to the CNN, we feed the input image to the CNN to generate a convolutional feature map. From the convolutional feature map, we identify the region of proposals and warp them into squares and by using a RoI pooling layer we reshape them into a fixed size so that it can be fed into a fully connected layer. From the RoI feature vector, we use a softmax layer to predict the class of the proposed region and also the offset values for the bounding box.
The reason “Fast R-CNN” is faster than R-CNN is because the convolution operation is done only once per image and a feature map is generated from it.

You Only Look Once (YOLO) :

YOLO or You Only Look Once is an object detection algorithm much different from the region based algorithms . In YOLO a single convolutional network predicts the bounding boxes and the class probabilities for these boxes.How YOLO works is that we take an image and split it into an SxS grid, within each of the grid we take m bounding boxes. For each of the bounding box, the network outputs a class probability and offset values for the bounding box. The bounding boxes having the class probability above a threshold value is selected and used to locate the object within the image.

Deconvolutional Neural Network/Deconv (DNNs):

A deconvolutional neural network is a neural network that performs an inverse convolution model. Some experts refer to the work of a deconvolutional neural network as constructing layers from an image in an upward direction, while others describe deconvolutional models as “reverse engineering” the input parameters of a convolutional neural network model.

Deconvolutional neural networks are also known as deconvolutional networks, deconvs or transposed convolutional neural networks.

Recurrent Neural Networks (RNNs) :

RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer.

RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer. The nodes in different layers of the neural network are compressed to form a single layer of recurrent neural networks.

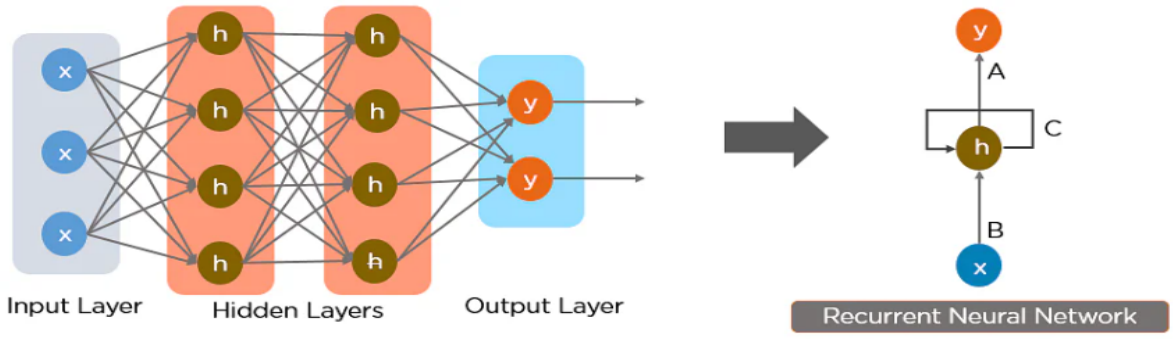


Fig: Simple Recurrent Neural Network

The input layer ‘x’ takes in the input to the neural network and processes it and passes it onto the middle layer.

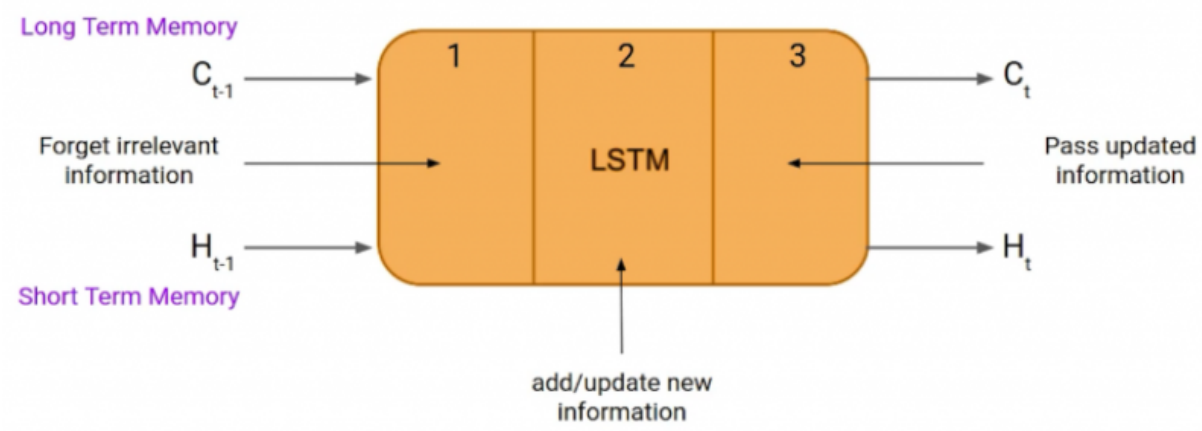
The middle layer ‘h’ can consist of multiple hidden layers, each with its own activation functions and weights and biases. If you have a neural network where the various parameters of different hidden layers are not affected by the previous layer, ie: the neural network does not have memory, then you can use a recurrent neural network.

The Recurrent Neural Network will standardize the different activation functions and weights and biases so that each hidden layer has the same parameters. Then, instead of creating multiple hidden layers, it will create one and loop over it as many times as required.

Long Short Term Memory (LSTM) :

Long Short Term Memory Networks is an advanced RNN, a sequential network, that allows information to persist. It is capable of handling the vanishing gradient problem faced by RNN.

At a high-level LSTM works very much like an RNN cell. Here is the internal functioning of the LSTM network. The LSTM consists of three parts, as shown in the image below and each part performs an individual function.



The first part chooses whether the information coming from the previous timestamp is to be remembered or is irrelevant and can

be forgotten. In the second part, the cell tries to learn new information from the input to this cell. At last, in the third part, the cell passes the updated information from the current timestamp to the next timestamp.These three parts of an LSTM cell are known as gates. The first part is called Forget gate, the second part is known as the Input gate and the last one is the Output gate.Just like a simple RNN, an LSTM also has a hidden state where H(t-1) represents the hidden state of the previous timestamp and Ht is the hidden state of the current timestamp. In addition to that LSTM also have a cell state represented by C(t-1) and C(t) for previous and current timestamp respectively.