Case Study VII

Krushang Shah

Prof. Richard Khan

Senior Design I

ID 23821856

Convolutional Neural Network

The name “Convolutional Neural Network” indicates that the network employs a mathematical operation called convolution. Convolution is a specialized kind of linear operation. Convolutional networks are simply neural networks that use convolution in the place of general matrix multiplication in at least one of their layers.

A Convolutional Neural Network consists of an input layer, lots of hidden layers and an output layer. The hidden layers consist of multiple layers that each work in a different way, to train the model on training data and make the model as accurate as possible. The different types of hidden layers that are present in a Convolutional Neural Network are

Convolutional Layer: This layer is used for convolution of the input, along the width and height of input volume, which gets better overtime, with each pass.

Local Connectivity: This layer relies on the correlation of the input, where it exploits the spatial local correlation, and each neuron of this layer is connected with a small region of input volume.

Spatial Arrangement: This layer depends on various factors like strides, depth and zero-padding, to determine the size of the output volume, which tells us the number of neurons in the output layer.

Dropout: This refers to the regularization technique used in neural networks, in order to reduce the overfitting of a model, by preventing complex co-adaptations on the training data. This is a very efficient way to perform model averaging on the neural network. From the name itself, it is understandable that it refers to the dropping out of units, both hidden and visible.

Parameter Sharing: This layer determines the number of free parameters in a neural network. This breaks the input into depth slices, where each slice uses same weights and biases, hence making the process faster and cheaper.

Pooling Layer: This layer takes the input and separates them in blocks as defined and then apples pooling on each block, to reduce the entire block into one cell. The pooling types that are usually used include max pooling, which takes the maximum element of the selected block for the next layer, and another type of pooling is average pooling, which takes the average of the entire block for the value of the cell for next layer.

ReLU Layer: ReLU stands for Rectified linear unit, which applies an activation function on the input, to take the maximum value, and eliminate all the negative values by setting them to zero.

Fully Connected Layer: This layer is connected to all activations of the previous layers. It is the layer after several convolutional and pooling layers, which does affine transformations and matrix multiplications, followed by a bias offset.

Loss Layer: This layer is used to check the loss that occurred during training, by checking the predicted labels and comparing them with true labels, to find the errors and the loss.

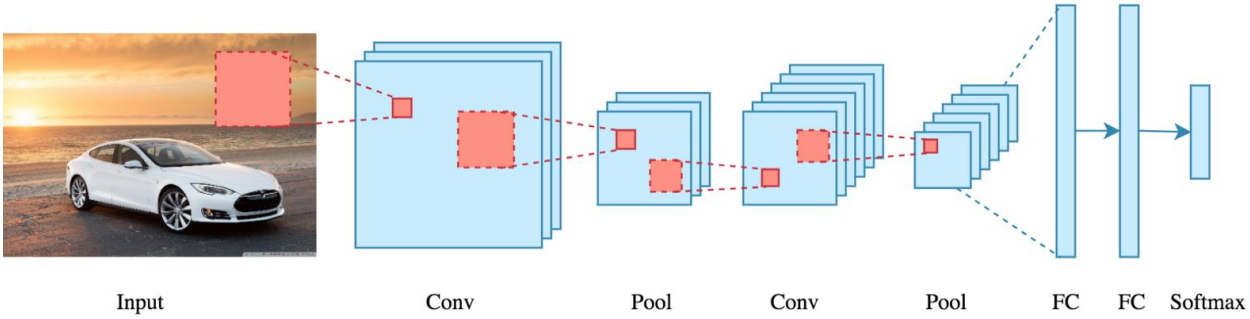


Figure 1: General example of a Convolutional Neural Network

The above shown diagram is a general example of how a neural network would look like. There are multiple hidden layers that are associated with the neural network. Each layer performs its specific task, and the final goal is to make the neural network as accurate as possible. The above neural network has an image of a deer as the input, which then passes through convolutional layer, to develop feature maps, for which each one is convoluted using different filters and filter biases. The next step shown in the process is pooling, which reduces the width and height of the feature maps by using max pooling, to get pooled feature maps which are smaller in width and height compared to feature maps. The next two steps are doing the convolution and pooling, to get a better and more accurate model. Then the next layer is fully connected layer, which is connected to the activation functions of all the previous layers to develop a neural network. And the final step is where we get classified output, for example in the model shown above, it predicts that the image is of a car, with a probability and the other classes also have some smaller probabilities. We finally apply SoftMax function, to set the value of the class that has maximum probability to one and all other classes to zero.

Dropout

The term Dropout refers to dropping of units both visible and hidden in a neural network. This is the most efficient way to prevent overfitting of a model. Overfitting means that the model is over trained on the training data, and hence, when exposed to the testing data, it performs badly. The best way of solving overfitting is applying a regularization technique. The regularization helps in reducing the overfitting, by applying a penalty to the loss function, which is the part of loss layer. The penalty is applied, so that the model does not learn the interdependent set of weights, so that it does not get over trained on the training data. This type of penalties is also known as Laplacian and Gaussian penalties in Logistic regression.

In a neural network, at each stage, individual nodes are either kept with a probability P, or dropped out with a probability of 1-P, so that a reduced network is left. The edges coming in and going out from a dropped-out node are also removed. The next stage of the neural network only gets the reduced network as the input for training. The removed nodes are then reinserted in the network with their original weights. Usually, the dropping probability of the hidden nodes is 0.5. For input nodes, the dropping probability should be much lower, otherwise the information coming in as input will be lost, without even being processed.

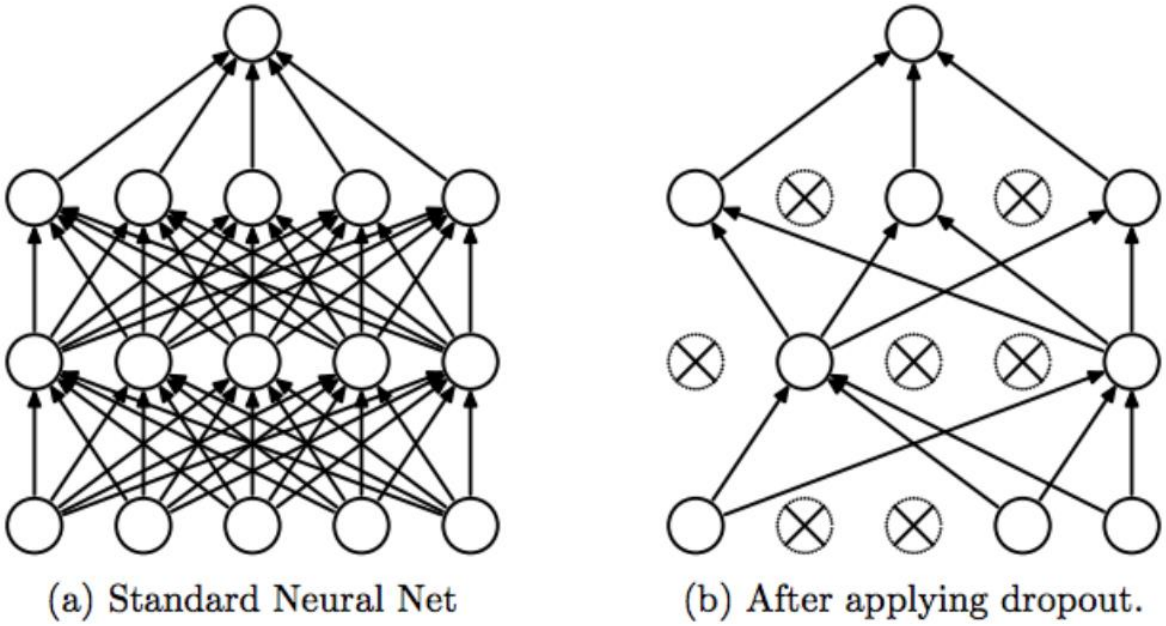
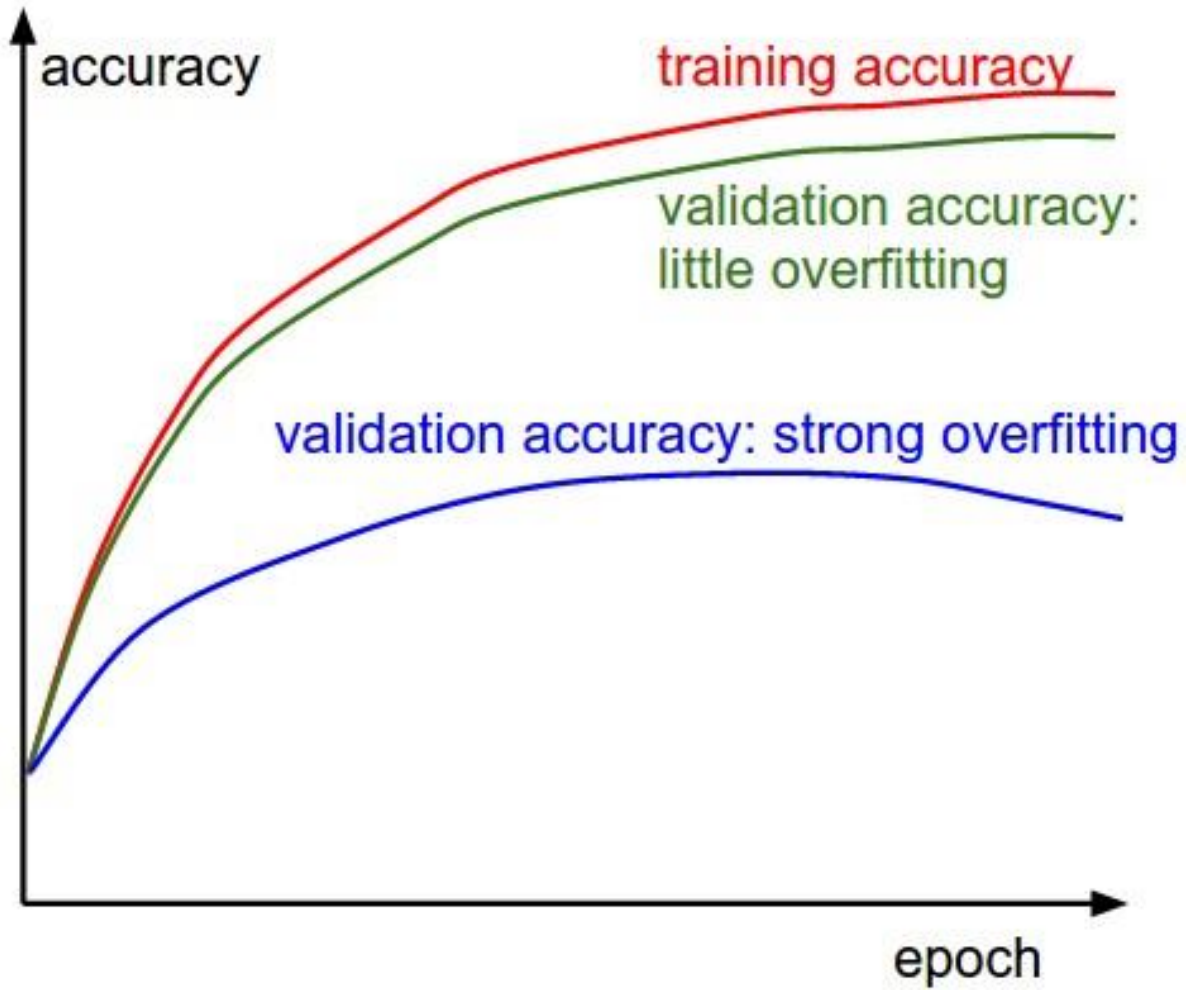


Figure 2: Difference between a fully connected neural network and network after applying dropout.

For the testing phase, ideally, we would want to get the sample average of all possible 2^n dropped-out networks, but it is not feasible for larger values of n. However, we can find the approximation by using the full network with each node's output weighted by a factor of P, so that the expected output value at each testing stage is same as that in the training stages. This is the most important contribution of the Dropout method. Even though we have almost 2^n neural nets in the training phase, for testing we only need a single neural network.

When we do not allow all nodes in the training data, this dropout of nodes reduces the overfitting. It also significantly improves the training speed and reduces the memory needed for training. This makes the combination of models easier even for deep neural networks. This technique reduces the node interactions, which leads them to more robust features that are better in the generalization of data.



The above shown image shows the difference between the accuracy of different types of the networks. The red line shows the training accuracy obtained while training the neural network on the training data. The green line is the accurate model obtained from testing on the dropout probability of 0.5 for hidden nodes, which has a little overfitting. The last one with the blue line is the one for validation accuracy, which had a fully connected network while training and hence, it got really accurate on the training data, but while testing data, it is performing poorly, because it expects relations between features from the input test data, which are not present in them.