**FAQs and Resources**

**Topics: Basics of Image handling and Introduction to Convolutional Neural Networks**

1. **What does the value at each pixel represent and why is the range from 0 to 255?**

The value at each pixel represents the intensity of the image at that location. For grey scale images it is a single integer value and for color images, it is a vector of 3 values each corresponding to different color (RGB) channels. The images we work with are generally 8-bit images. Each pixel value is stored in 8-bit values. The highest value an 8-bit number can hold is 255 and the lowest value is 0. 16-bit and higher bit images also exist which have a finer resolution of intensity, but they are not used in day to day images.

1. **What is the major difference between fully connected neural networks and a convolutional neural network?**

Each neuron in a fully connected neural network layer is connected to all the neurons in the previous layer. Hence, the name ‘fully connected’ neural network. Whereas in a convolutional neural network each neuron in a convolutional layer is connected to only 4(2X2) or 9(3X3) or other smaller number of neurons in the previous layer depending on the size of the kernel. Also, multiple neurons in the convolutional layer share the same weights/parameters unlike in fully connected layers. So the number of parameters in a convolutional layer is much less when compared to a fully connected layer with the same number of neurons.

1. **Why can’t we use fully-connected neural networks for images?**

Images are getting larger and larger every day with new cameras and compact storage capacities. Larger images mean a larger number of input values. As for the same reason specified in the answer to Question 2, the number of learnable parameters increases exponentially with the size of the input in fully connected layers and it gets difficult to compute backpropagation. Also, the kernels/filters learned in a convolutional neural network try to learn features which are spatially invariant. So, a pattern ‘X’ in the top left corner of the image, or the same pattern in the bottom right corner of the image can both be detected by a single kernel/filter. The features learned by fully connected neural networks are not spatially invariant.

1. **Why do we need GPUs to work with convolutional neural networks?**

To train an image based classifier, you need thousands of images for each class. Even though convolutional neural networks don’t need as many parameters as fully connected neural networks, they still have a lot of parameters to train. To be able to compute so many outputs and gradients for all the thousands of images, we require parallel computing capability to speed up the learning process. At the end of compiling any model in Keras. Use model.summary() function to visualize the number of parameters to be trained next time.

1. **Do we use bias in convolutional neural networks too?**

Yes, Bias is also used in convolutional layers. We don’t explicitly write it but while computing the output of a neuron we also add a bias to the weighted sum from the kernel before passing it through an activation function. But, there is no concept of bias in pooling layers. There are no learnable parameters in pooling layers.

1. **Why do convolutional neural networks work better than the existing handcrafted filters at detecting features?**

First of all, handcrafted features are difficult to design for each different problem. Most of the handcrafted kernels are based on mathematical distributions or transformations. These kernels don’t necessarily work with all sorts of images. The real world images are complex and do not fall into any particular probability distribution. Convolutional neural networks learn kernels/features specific to the problem at hand. The disadvantage is that one cannot control the features being learned by the network. CNN can learn complex features which are incomprehensible by humans but, still work for the task they are learned for.

1. **For training a CNN classifier is it necessary to have an equal number of training examples for each class?**

It is advised to have equal number of training examples for each class. Otherwise, class

Imbalance problems occur while training the neural network. When the ratio of the number of training samples for each class is skewed, it is reflected in each batch of inputs fed to the neural network during forward propagation. Because of this, the gradients of the dominant class samples will over power the gradients of the other classes in a batch. As batch gradient descent sums all the gradients in a batch, the gradients of the major class samples dominate the whole batch gradient and result in poor performance for minor classes.

1. **Do the pooling layers contribute in the back propagation even though they don’t have any learnable parameters?**

Yes, even though there are no learnable parameters in the pooling layer it affects the backpropagation. While back propagating the gradient through the max pooling layer, except for the neuron which had the maximum activation the rest of the neurons get zero gradient as their outputs do not change the error. The neuron which had the maximum activation gets copied with the gradient from the max pooling layer. In average pooling, the gradient is divided equally among all the contributing neurons.

1. **Where to find datasets to work with?**

[Kaggle](https://www.kaggle.com/) is one of the best sources for the latest datasets and competitions in deep learning. You can find a lot of datasets from different organizations and companies on a variety of topics on Kaggle. [UCI](https://archive.ics.uci.edu/ml/index.php) ML Repository is also a good database for a lot of image and other datasets. [AWS Open data](https://registry.opendata.aws/) is a dataset repository by Amazon. Apart from these, there are some very common datasets like [Youtube 8M](https://research.google.com/youtube8m/), [COCO](http://cocodataset.org/#home), [Pascal](http://pascallin.ecs.soton.ac.uk/challenges/VOC/), [Imagenet](http://www.image-net.org/), [CIFAR 10](http://www.cs.utoronto.ca/%7Ekriz/cifar.html), [Caltech 256](http://www.vision.caltech.edu/Image_Datasets/Caltech256/%29%C2%A0), [Open Images Dataset](https://storage.googleapis.com/openimages/web/index.html). Most of the papers try to evaluate their models on these common datasets so that they can show a clear comparison of performance with other models.

1. **What are some good practices of collecting image datasets?**

Dataset collection is one of the key aspects of deep learning. If the dataset is skewed, there are high chances that the trained model would also be skewed. There should always be sufficient amount testing data for any problem. And the images in the test data should never occur in the training set as it would defeat the whole purpose of having a test set. Both the training set and the test should try to encompass all the variations of inputs and outputs. For an example, lets say an image dataset of cats and dogs is being collected. If all the cat images are captured during night time and all the dog images are captured during day time. The model trained on this dataset might not learn the features required for cats and dogs but may try to learn the features to differentiate between night and day. To eliminate these errors, all the scenarios should be carefully considered and decided before collecting dataset. The test set of images should be able to give a correct measure of the effectiveness of the model.

1. **How to get codes/weights for different architectures and how to find out the state of the art architectures for each problem?**

Most of the new architectures/papers publish their papers on [arXiv](https://arxiv.org/). You can access their papers freely on arXiv. Reading the papers would provide a good idea of how the architectures have been implemented. Many papers also include the github url of the model code for reproducing the results. [PapersWithCode](https://paperswithcode.com/) is a good place to search for different papers which also provide code. It also shows the latest state-of-the-art architecture for each different problem in the computer vision field [here](https://paperswithcode.com/area/computer-vision). You can also browse the top architectures for each popular dataset like Pascal, COCO etc., on it. Apart from this, you can also try to search for the implementation of an architecture on [Github](https://github.com/).