← Go Back to Deep Learning

Course Content

Transfer Learning

In the previous two lectures, we discussed the working of different types of neural networks from scratch.
While training models from scratch has advantages in terms of customizability, it also has significant disadvantages in terms of the computational resources required to do the same, especially for deep neural networks that need to be trained on a lot of data.

Breakthroughs in deep learning require huge amounts of data, which can be costly and sometimes infeasible. It can also take a lot of research and effort to come up with the perfect architecture that gives good results.

So, what can be done if the data is scarce? This is where the method of **Transfer Learning** comes into play.

Transfer Learning is the process of using a pre-trained model, which is trained on a prior task (called the **pretext task**) similar to the task at hand, called the **target task**.

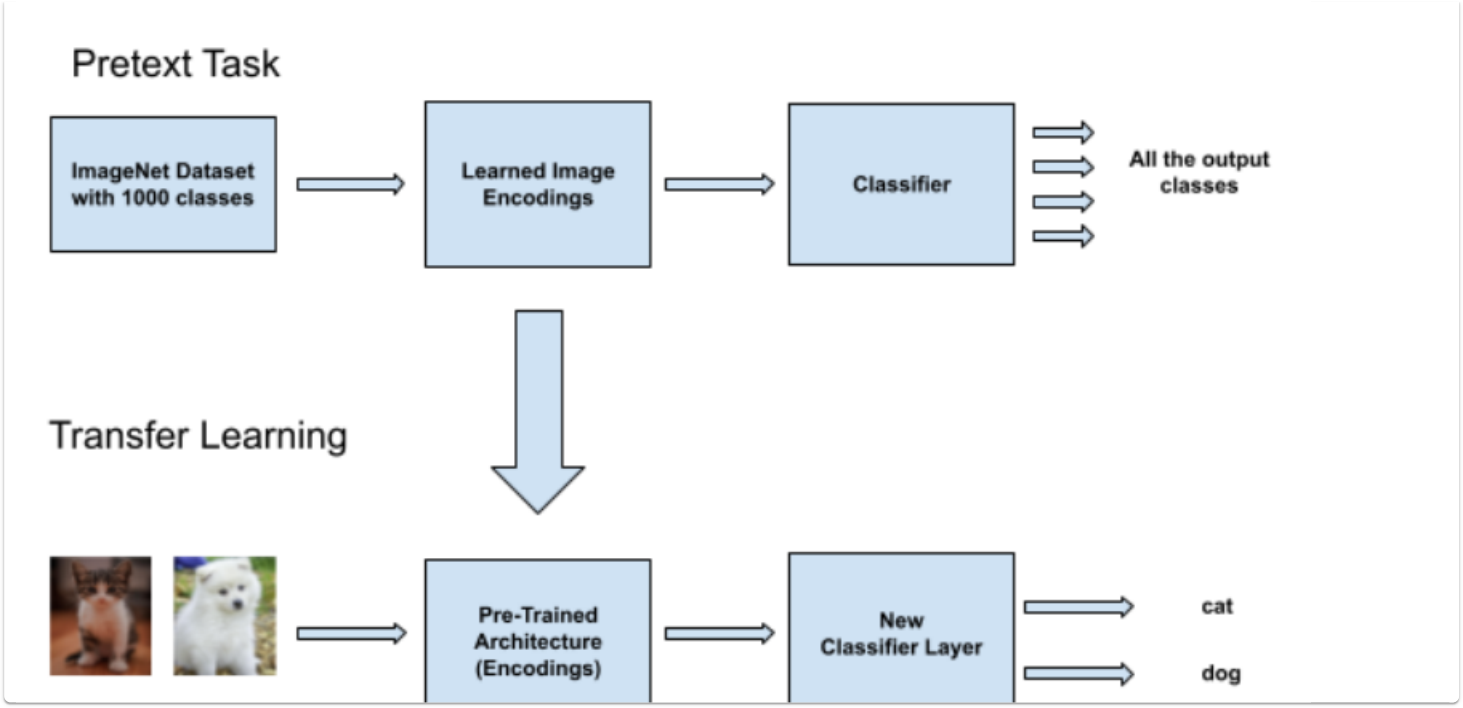
Let's understand this with an example:

Suppose we want to build a dog/cat classifier but we lack the volume of data and computational resources required. One way to achieve this task would be to use a model that is already trained on the same or a similar task with a good performance. A similar task to the problem of classifying dogs and cats is the **ImageNet** problem statement.

ImageNet is a famous multi-class image classification challenge among deep learning practitioners where the task is to come up with a model that can classify the input image data into one of 1,000 output classes, with very high accuracy. VGG16, which has been one of the historical standout neural network architectures in this challenge, is a popular pre-trained neural network model. So in our example, we can use the weights of the VGG16 model that it has already learned by training on thousands of labeled images from ImageNet. The advantage of doing so is that since VGG16 is a State of the Art model which has been developed with lots of research and development, it is always more likely that the pre-trained architecture will be able to give good performance.

Now, our problem at hand has only two classes, i.e., cats and dogs. Hence, we take the model of VGG16 as it is and we just re-train the classification layer to have two output neurons only. So, during the training process, we set the weights of all the previous layers as non-trainable and only set the last output layers as trainable.

The image below shows the flow of the Transfer Learning model for the example discussed above:



Help