Transfer Learning

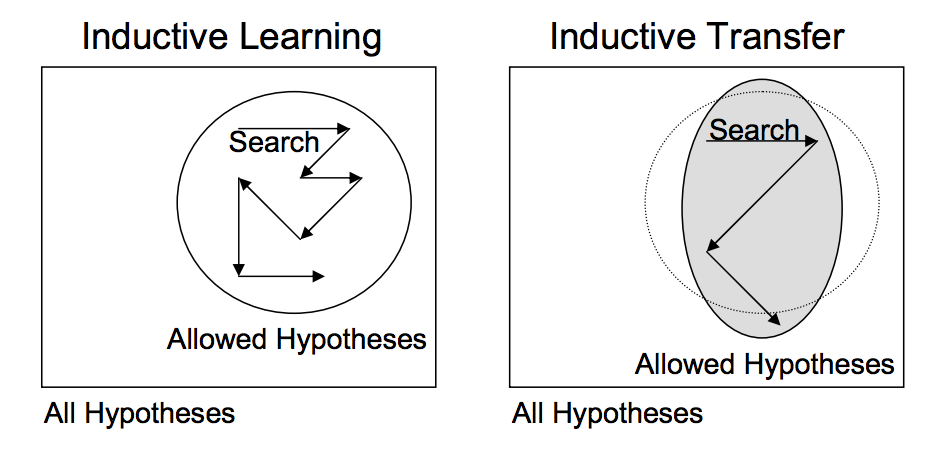
Submitted by- Sandeep

Date- 28th Oct, 2020

**Transfer learning** is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.

It is a popular approach in deep learning where **pre-trained models** are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems.

Transfer learning only works in deep learning if the model features learned from the first task are general.This form of transfer learning used in deep learning is called inductive transfer. This is where the scope of possible models (model bias) is narrowed in a beneficial way by using a model fit on a different but related task.



## 

## **How to Use Transfer Learning?**

Two common approaches are as follows:

1. Develop Model Approach
2. Pre-trained Model Approach

### 

### **Develop Model Approach**

1. Select Source Task. You must select a related predictive modeling problem with an abundance of data where there is some relationship in the input data, output data, and/or concepts learned during the mapping from input to output data.
2. Develop Source Model. Next, you must develop a skillful model for this first task. The model must be better than a naive model to ensure that some feature learning has been performed.
3. Reuse Model. The model fit on the source task can then be used as the starting point for a model on the second task of interest. This may involve using all or parts of the model, depending on the modeling technique used.
4. Tune Model. Optionally, the model may need to be adapted or refined on the input-output pair data available for the task of interest.

### **Pre-trained Model Approach**

1. Select Source Model. A pre-trained source model is chosen from available models. Many research institutions release models on large and challenging datasets that may be included in the pool of candidate models from which to choose from.
2. Reuse Model. The pre-trained model can then be used as the starting point for a model on the second task of interest. This may involve using all or parts of the model, depending on the modeling technique used.
3. Tune Model. Optionally, the model may need to be adapted or refined on the input-output pair data available for the task of interest.

This second type of transfer learning is common in the field of deep learning. [Source](https://machinelearningmastery.com/transfer-learning-for-deep-learning/)

We have used the **second** form of transfer learning.

### **Why do we usually choose image size as a [224,224] square?**

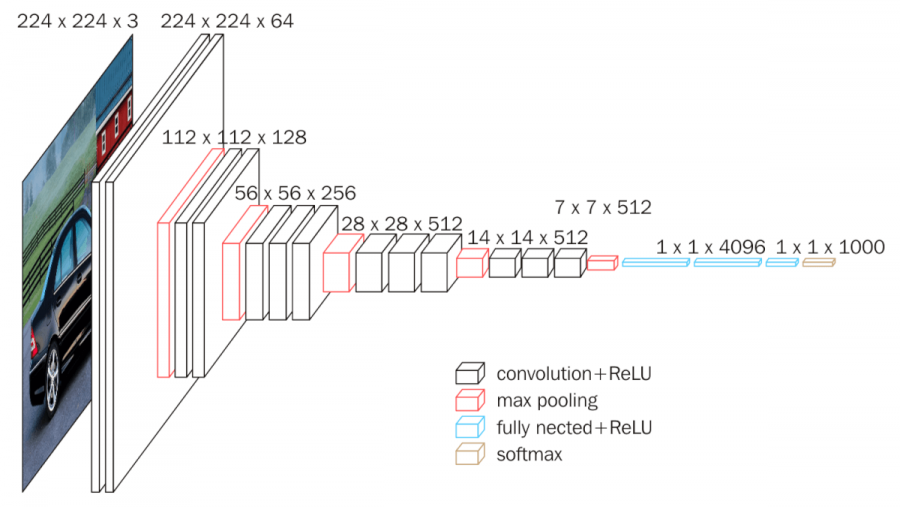
It is **not** compulsory to have a [224,224] image size for a **convolutional neural network** to function normally.It is chosen for programmatic reasons such as a compromise between using image details vs number of parameters and training set size required.

In our case we were using **VGG16** and we have used images of different resolutions to train our model, then taking a **square crop** from the middle could be a reasonable compromise.



# VGG16

**VGG16** is a convolutional neural network model ,the model achieves **92.7%** top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.



The input to cov1 layer is of fixed size 224 x 224 RGB image. The image is passed through a stack of convolutional (conv.) layers, where the filters were used with a very small receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, center). In one of the configurations, it also utilizes 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2 pixel window, with stride 2.

Unfortunately, there are two major drawbacks with VGGNet:

1. It is painfully slow to train.(which we have experienced)
2. The network architecture weights themselves are quite large (concerning disk/bandwidth).

## What did I do?

I trained only the last layer of the model from scratch(with a dataset <1000) and obtained an increasing accuracy of 82.3% in just 5 epochs and an incrementing curve.

[Link](https://github.com/Sandeep19531/Transfer_Learning)

## Bibliography

**Krish Naik** for helping me understand this topic and also for sharing his github repository for giving me an idea of the practical implementation of the [code](https://github.com/krishnaik06/Transfer-Learning)

[Aqeel Anwar](https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96) for his blog on ‘Difference between AlexNet, VGGNet, ResNet, and Inception’.

[Jason Brownlee](https://machinelearningmastery.com/how-to-improve-performance-with-transfer-learning-for-deep-learning-neural-networks/) for his blog on ‘How to Improve Performance With Transfer Learning for Deep Learning Neural Networks’.

**Thank you** for engaging me into such an amazing project.