Assignment No.6

Aim:. To implement Object detection using Transfer Learning of CNN architectures

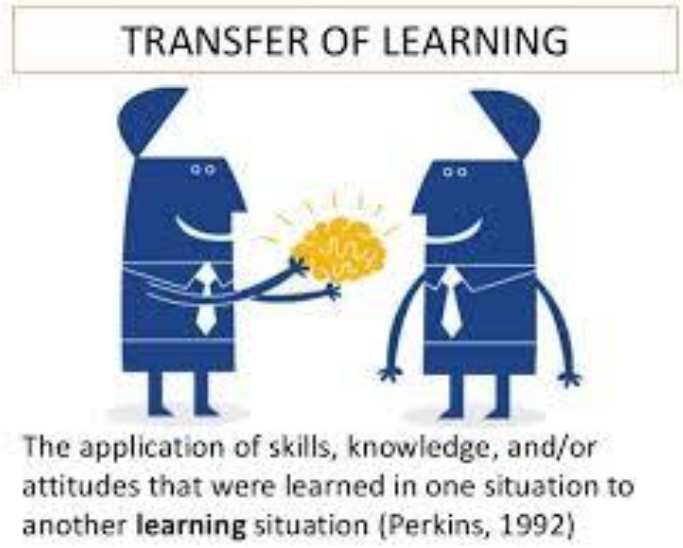
Objectives:

1. Study of different architectures of CNN.
2. Study of transfer learning and object detection.
3.perfom following tasks
    a. Load in a pre-trained CNN model trained on a large dataset
    b. Freeze parameters (weights) in model's lower convolutional layers
    c. Add custom classifier with several layers of trainable parameters to model
    d. Train classifier layers on training data available for task
    e. Fine-tune hyper parameters and unfreeze more layers as needed

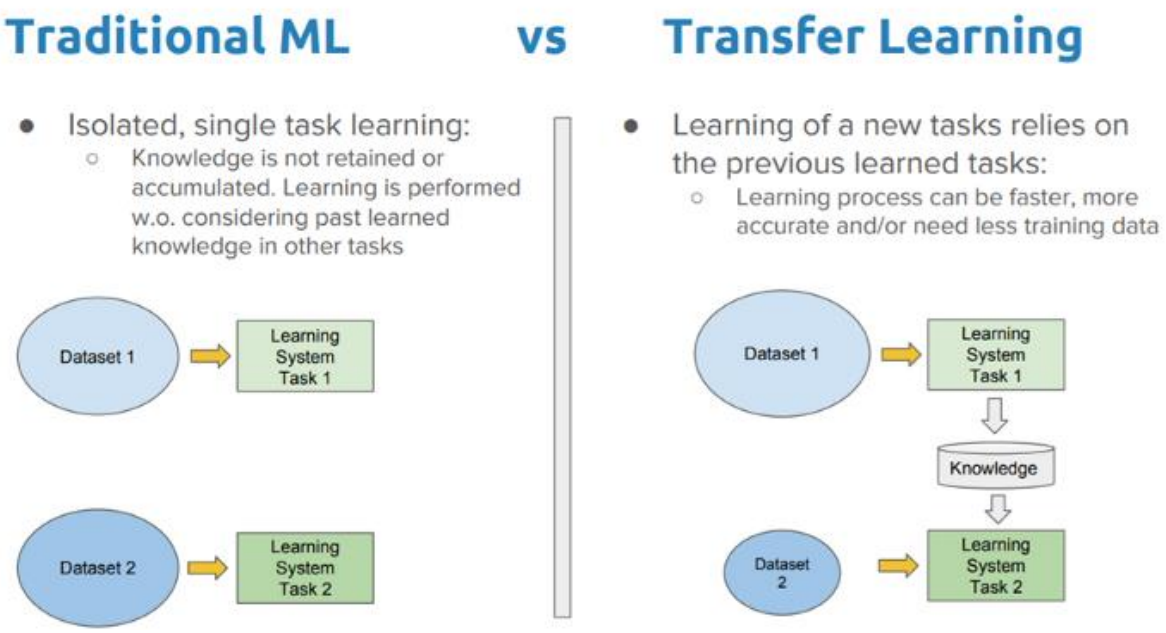
Theory:

What is transfer learning?

A teacher has years of experience in the particular topic he/she teaches. With all this accumulated information, the lectures that students get is a concise and brief overview of the topic. So it can be seen as a "transfer" of information from the learned to a novice.



A neural network is trained on a data. This network gains knowledge from this data, which is compiled as "weights" of the network. These weights can be extracted and then transferred to any other neural network. Instead of training the other neural network from scratch, we **"transfer"** the learned features.



What is a Pre-trained Model?

A pre-trained model is a model created by someone else to solve a similar problem. Instead of building a model from scratch to solve a similar problem, you use the model trained on other problem as a starting point.

Transfer learning is usually done for tasks where your dataset has too little data to train a full-scale model from scratch.

The most common incarnation of transfer learning in the context of deep learning is the following workflow:

1. Take layers from a previously trained model.
2. Freeze them, so as to avoid destroying any of the information they contain during future training rounds.
3. Add some new, trainable layers on top of the frozen layers. They will learn to turn the old features into predictions on a new dataset.
4. Train the new layers on your dataset.

A last, optional step is **fine-tuning**, which consists of unfreezing the entire model you obtained above (or part of it), and re-training it on the new data with a very low learning rate. This can potentially achieve meaningful improvements, by incrementally adapting the pretrained features to the new data.

Fine-tuning

Once your model has converged on the new data, you can try to unfreeze all or part of the base model and retrain the whole model end-to-end with a very low learning rate.

This is an optional last step that can potentially give you incremental improvements. It could also potentially lead to quick overfitting -- keep that in mind.

It is critical to only do this step *after* the model with frozen layers has been trained to convergence. If you mix randomly initialized trainable layers with trainable layers that hold pre-trained features, the randomly-initialized layers will cause very large gradient updates during training, which will destroy your pre-trained features.

It's also critical to use a very low learning rate at this stage, because you are training a much larger model than in the first round of training, on a dataset that is typically very small. As a result, you are at risk of overfitting very quickly if you apply large weight updates. Here, you only want to readapt the pretrained weights in an incremental way.

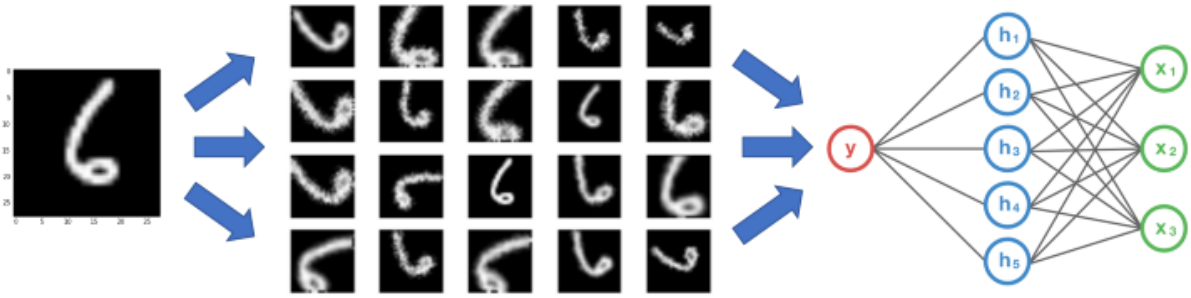
Data augmentation

Data augmentation is a process of artificially increasing the amount of data by generating new data points from existing data. This includes adding minor alterations to data or using machine learning models to generate new data points in the latent space of original data to amplify the dataset.

Minor changes such as flips or translations or rotations. Our neural network would think these are distinct images anyway.

A question may arise about the difference between augmented data and synthetic data.

- **Synthetic data**: When data is generated artificially without using real-world images. Synthetic data are often produced by Generative Adversarial Networks
- **Augmented data**: Derived from original images with some sort of minor geometric transformations (such as flipping, translation, rotation, or the addition of noise) in order to increase the diversity of the training set.



A convolutional neural network that can robustly classify objects even if its placed in different orientations is said to have the property called **invariance**. More specifically, a CNN can be invariant to **translation, viewpoint, size** or **illumination** (Or a combination of the above).

This essentially is the premise of **data augmentation**

Algorithms/steps

Transfer learning & fine-tuning
▻ Setup
▻ Introduction
▻ Freezing layers: understanding the trainable attribute
▻ Recursive setting of the trainable attribute
▻ The typical transfer-learning workflow
▻ Fine-tuning
▻ Transfer learning & fine-tuning with a custom training loop
▻ An end-to-end example: fine-tuning an image classification model on a cats vs. dogs dataset
Getting the data
Standardizing the data
Using random data augmentation
▻ Build a model
▻ Train the top layer
▻ Do a round of fine-tuning of the entire model

Conclusion

Thus we have learned what is transfer learning, pretrained model and fine tuning

OUTPUT