**CSE499A.10 - Transfer Learning**

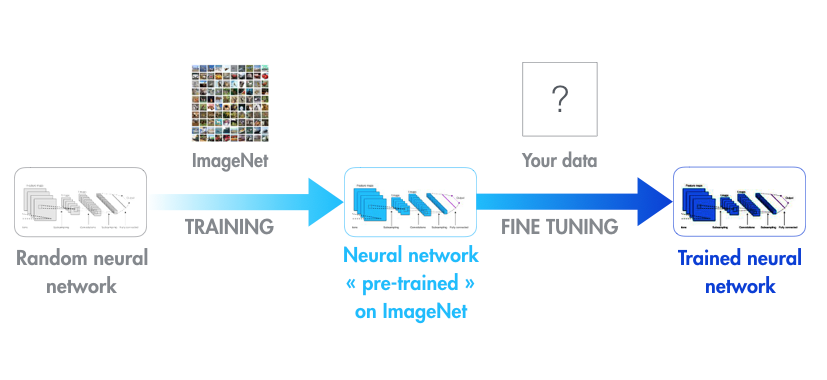
We got an abstract idea about Transfer Learning. I’m going to present what we have learning so far.

So, Transfer learning is the use of the knowledge gained while solving one problem and applying it to a different but related problem.

For example, we train a model for recognizing cars, with some fine tuning that model can be used to some extent to recognize trucks.

The Idea behind Transfer Learning is that instead of training a deep network from scratch

* We take a network trained on a different domain for a different source task
* And adapt it for our domain for our targeted task

****

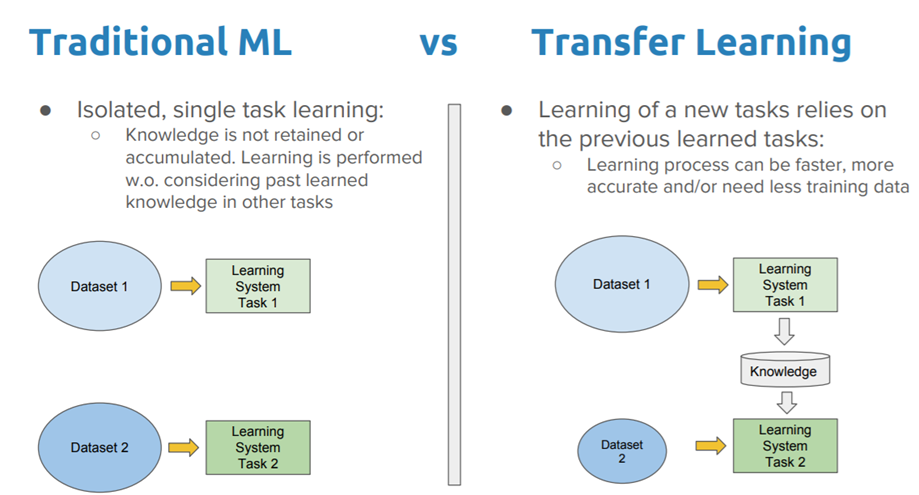
**If we look at the picture:**

A random neural network is taken and trained on a large dataset (Like ImageNet). All the parameters of the neural network are trained and therefore the model is learned. It may take hours on our GPU.

Then we give our dataset to fine tune that pre-trained network. Considering that the new dataset is almost similar to the original dataset.

The neural network does not need to learn from scratch to discover basic shapes and textures, only the highest-level layers need to be updated to adjust to our task. And that is Transfer Learning.

1. *If the new dataset is very small, it’s better to train only the final layers of the network to avoid overfitting,* *keeping all other layers fixed. So, remove the final layers of the pre-trained network. Add new layers.****Retrain only the new layers.***
2. ***If the new dataset is very much large, retrain the whole network****with initial weights from the pretrained model.*



* In traditional ML, learning is performed without considering past learned knowledge in other tasks.
* But in Transfer Learning, learning of a new task relies on the previous tasks. So, learning process can be faster, more accurate and need less training data.

**There are two approaches to Transfer Learning:**

**1. Training A Model to Reuse It:**

Imagine we want to solve task A but don’t have enough data to train a deep neural network. One way around this is to find a related task B with an abundance of data. Train the deep neural network on task B and use the model as a starting point for solving task A. Whether we'll need to use the whole model or only a few layers depend heavily on the problem we're trying to solve.

**2. Using A Pre-Trained Model:**

The second approach is to use an already pre-trained model. Keras, for example, provides nine pre-trained models that can be used for transfer learning, prediction, feature extraction and fine-tuning.

Xception, VGG16, VGG19, ResNet, ResNetV2, InceptionV3, InceptionResNetV2, MobileNet, MobileNetV2, DenseNet, NASNet

**Popular Pre-Trained Models:**

* There are some pre-trained machine learning models out there that are quite popular. One of them is the Inception-v3 model, which was trained for the ImageNet “Large Visual Recognition Challenge." In this challenge, participants had to classify images into 1,000 classes like “zebra," “Dalmatian" and “dishwasher."
* Microsoft also offers some pre-trained models, available for both R and Python development, through the MicrosoftML R package and the Microsoftml Python package.
* Other quite popular models are ResNet (Short for Residual Network. most popularly used for image classification)
* And AlexNet (a significantly “old” image classification algorithm that performs well on ImageNet.)

**Why Transfer Learning?**

Transfer learning is useful when we have insufficient data for a new domain we want handled by a neural network and there is a big pre-existing data pool that can be transferred to our problem.

So, we might have only 1,000 images, but by tapping into an existing CNN such as ResNet, trained with more than 1 million images, we can gain a lot of low-level and mid-level feature definitions.

Transfer learning will become a key driver of Machine Learning success in industry.