ML | Introduction to Transfer Learning

We, humans, are very perfect at applying the transfer of knowledge between tasks. This means that whenever we encounter a new problem or a task, we recognize it and apply our relevant knowledge from our previous learning experiences. This makes our work easy and fasts to finish. For instance, if you know how to ride a bicycle and if you are asked to ride a motorbike which you have never done before. In such a case, our experience with a bicycle will come into play and handle tasks like balancing the bike, steering, etc. This will make things easier compared to a complete beginner. Such learnings are very useful in real life as it makes us more perfect and allows us to earn more experience. Following the same approach, a term was introduced *Transfer Learning* in the field of machine learning. This approach involves the use of knowledge that was learned in some task and applying it to solve the problem in the related target task. While most machine learning is designed to address a single task, the development of algorithms that facilitate transfer learning is a topic of ongoing interest in the machine-learning community.

Transfer learning is a technique in machine learning where a model trained on one task is used as the starting point for a model on a second task. This can be useful when the second task is similar to the first task, or when there is limited data available for the second task. By using the learned features from the first task as a starting point, the model can learn more quickly and effectively on the second task. This can also help to prevent overfitting, as the model will have already learned general features that are likely to be useful in the second task.

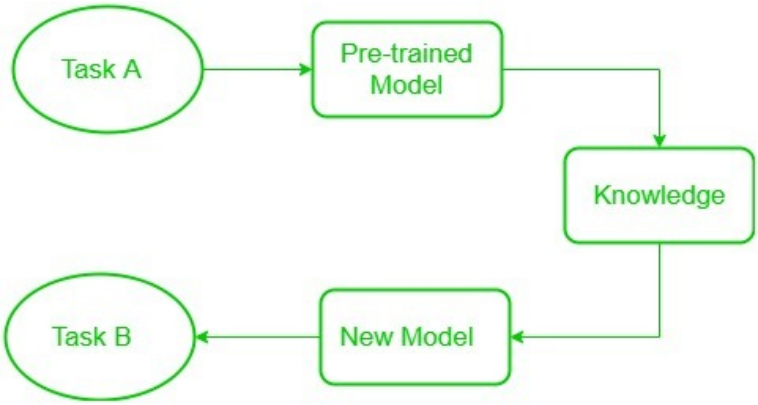
Why transfer learning?

Many deep neural networks trained on images have a curious phenomenon in common: in the early layers of the network, a deep learning model tries to learn a low level of features, like detecting edges, colors, variations of intensities, etc. Such kind of features appears not to be specific to a particular dataset or a task because no matter what type of image we are processing either for detecting a lion or car. In both cases, we have to detect these low-level features. All these features occur regardless of the exact cost function or image dataset. Thus learning these features in one task of detecting lions can be used in other tasks like detecting humans.

Necessity for transfer learning: Low-level features learned for task A should be beneficial for learning of model for task B.

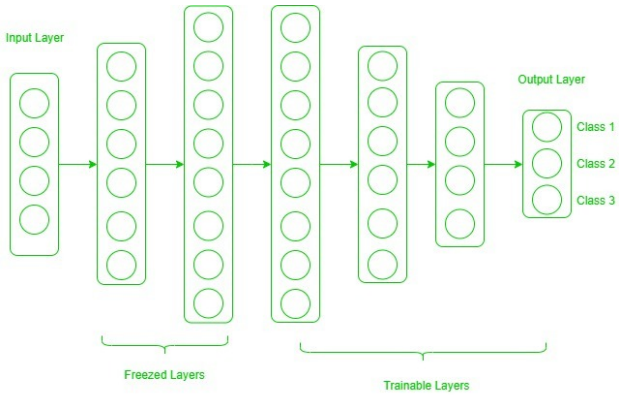
This is what transfer learning is. Nowadays, it is very hard to see people training whole convolutional neural networks from scratch, and it is common to use a pre-trained model trained on a variety of images in a similar task, e.g models trained on ImageNet (1.2 million images with 1000.

The Block diagram is shown below as follows:



categories), and use features from them to solve a new task. When dealing with transfer learning, we come across a phenomenon called the freezing of layers. A layer, it can be a CNN layer, hidden layer, a block of layers, or any subset of a set of all layers, is said to be fixed when it is no longer available to train. Hence, the weights of freeze layers will not be updated during training. While layers that are not frozen follows regular training procedure. When we use transfer learning in solving a problem, we select a pre-trained model as our base model. Now, there are two possible approaches to using knowledge from the pre-trained model. The first way is to freeze a few layers of the pre-trained model and train other layers on our new dataset for the new task. The second way is to make a new model, but also take out some features from the layers in the pre-trained model and use them in a newly created model. In both cases, we take out some of the learned features and try to train the rest of the model. This makes sure that the only feature that may be the same in both of the tasks is taken out from the pre-trained model, and the rest of the model is changed to fit the new dataset by training.

Froze and Trainable Layers:



Now, one may ask how to determine which layers we need to freeze and which layers need to train. The answer is simple, the more you want to inherit features from a pre-trained model, the more you have to freeze layers. For instance, if the pre-trained model detects some flower species and we need to detect some new species. In such a case, a new dataset with new species contains a lot of features similar to the pre-trained model. Thus, we freeze less number of layers so that we can use most of its knowledge in a new model. Now, consider another case, if there is a pre-trained model which detects humans in images, and we want to use that knowledge to detect cars, in such a case where the dataset is entirely different, it is not good to freeze lots of layers because freezing a large number of layers will not only give low level features but also give high-level features like nose, eyes, etc which are useless for new dataset (car detection). Thus, we only copy low-level features from the base network and train the entire network on a new dataset. Let's consider all situations where the size and dataset of the target task vary from the base network.

The target dataset is small and similar to the base network dataset: Since the target dataset is small, that means we can fine-tune the pre-trained network with the target dataset. But this may lead to a problem of overfitting. Also, there may be some changes in the number of classes in the target task. So, in such a case we remove the fully connected layers from the end, maybe one or two, and add a new fully-connected layer satisfying the number of new classes. Now, we freeze the rest of the model and only train newly added layers.

The target dataset is large and similar to the base training dataset: In such cases when the dataset is large and it can hold a pre-trained model there will be no chance of overfitting. Here, also the last full-connected layer is removed, and a new fully-connected layer is added with the proper number of classes. Now, the entire model is trained on a new dataset. This makes sure to tune the model on a new large dataset keeping the model architecture the same.

The target dataset is small and different from the base network dataset: Since the target dataset is different, using high-level features of the pre-trained model will not be useful. In such a case, remove most of the layers from the end in a pre-trained model

model and train the rest of the layers to fit a new dataset. Sometimes, it is beneficial to train the entire network after adding a new layer at the end.

The target dataset is large and different from the base network dataset: Since the target network is large and different, the best way is to remove the last layers from the pre-trained network and add layers with a satisfying number of classes, then train the entire network without freezing any layer.

Transfer learning is a very effective and fast way, to begin with, a problem. It gives the direction to move, and most of the time best results are also obtained by transfer learning.

*Below is the sample code using Keras for **Transfer learning & fine-tuning with a custom training loop.***

Advantages :

- Speed up the training process: By using a pre-trained model, the model can learn more quickly and effectively on the second task, as it already has a good understanding of the features and patterns in the data.
- Better performance: Transfer learning can lead to better performance on the second task, as the model can leverage the knowledge it has gained from the first task.
- Handling small datasets: When there is limited data available for the second task, transfer learning can help to prevent overfitting, as the model will have already learned general features that are likely to be useful in the second task.

Disadvantages:

Disadvantages of transfer learning:

- **Domain mismatch:** The pre-trained model may not be well-suited to the second task if the two tasks are vastly different or the data distribution between the two tasks is very different.
- **Overfitting**: Transfer learning can lead to overfitting if the model is fine-tuned too much on the second task, as it may learn task-specific features that do not generalize well to new data.
- **Complexity**: The pre-trained model and the fine-tuning process can be computationally expensive and may require specialized hardware.

Reference:

A popular reference book for transfer learning is “Transfer Learning for Computer Vision: Techniques and Applications” by Mohammad Raza Shah and Mubarak Shah. This book provides an overview of transfer learning techniques and their applications in computer vision, including detailed explanations of different types of transfer learning, such as instance-based, feature-based, and relational transfer learning. It also includes case studies and examples of transfer learning in real-world applications, such as image classification, object detection, and facial recognition. This book is a valuable resource for researchers, engineers, and practitioners working in computer vision and machine learning.

Last Updated : 09 Mar, 2023

Similar Reads

Deep Transfer Learning - Introduction

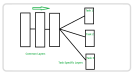
ML | Transfer Learning with Convolutional Neural Networks

Multiclass image classification using Transfer learning

Lung Cancer Detection Using Transfer Learning

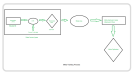
Dog Breed Classification using Transfer Learning

Transfer Learning with Fine-tuning in NLP

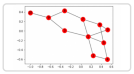


Introduction to Multi-Task Learning(MTL) for Deep Learning

Artificial intelligence vs Machine Learning vs Deep Learning



Meta-Learning in Machine Learning

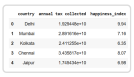


ML | Reinforcement Learning Algorithm : Python Implementation using Q-learning

Related Tutorials

Computer Vision Tutorial

Computer Science and Programming For Kids



Pandas AI: The Generative AI Python Library

Top Computer Vision Projects (2023)

Deep Learning Tutorial

Previous

Image Classifier using CNN

Next

Introduction to Recurrent Neural Network

Article Contributed By :

VikashChouhan

VikashChouhan

Follow

Vote for difficulty

Current difficulty : Medium

Easy | Normal | Medium | Hard | Expert

Improved By : jatingrg2399, chandanasamineni23, rathodavinash1181

Article Tags : Neural Network , Computer Subject , Machine Learning

Practice Tags : Machine Learning

Improve Article | Report Issue

GeeksforGeeks

A-143, 9th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org