

Contents

Transfer Learning Models	2
LeNet.....	2
AlexNet.....	3
VGGNET.....	4
RezNet.....	5
Other good source for more information about transfer learning.....	6

This doc is my comprehending and summarize of below course:

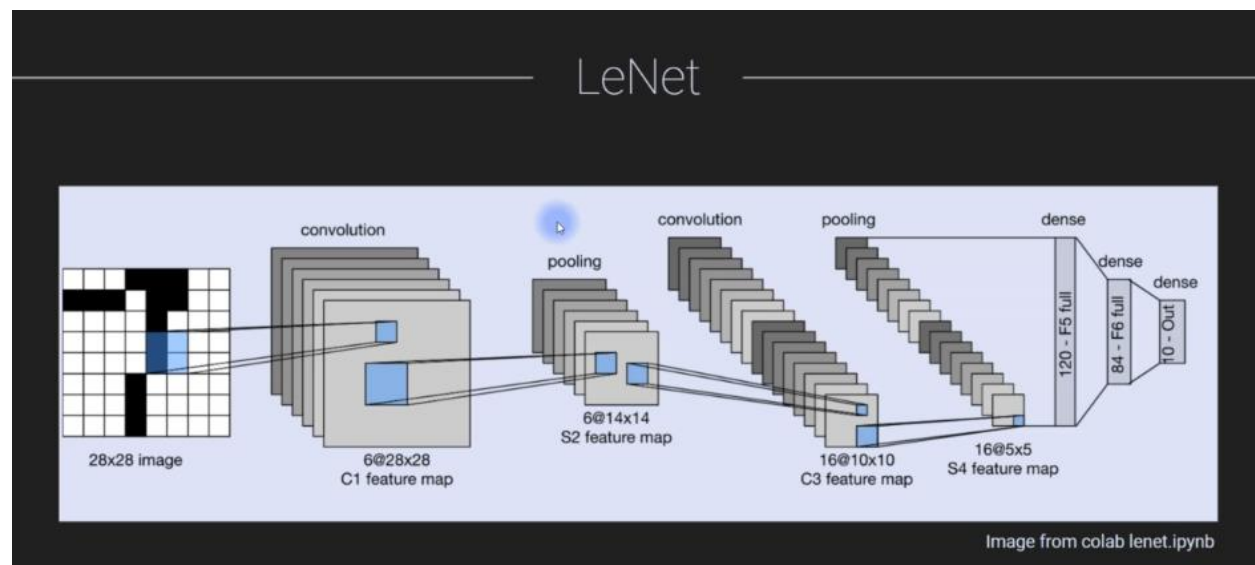
COURSE: A deep understanding of deep learning
SECTION: Transfer learning
TEACHER: Mike X Cohen, sincxpress.com
COURSE URL: [udemy.com/course/dudl/?couponCode=202108](https://www.udemy.com/course/dudl/?couponCode=202108)

Transfer Learning models

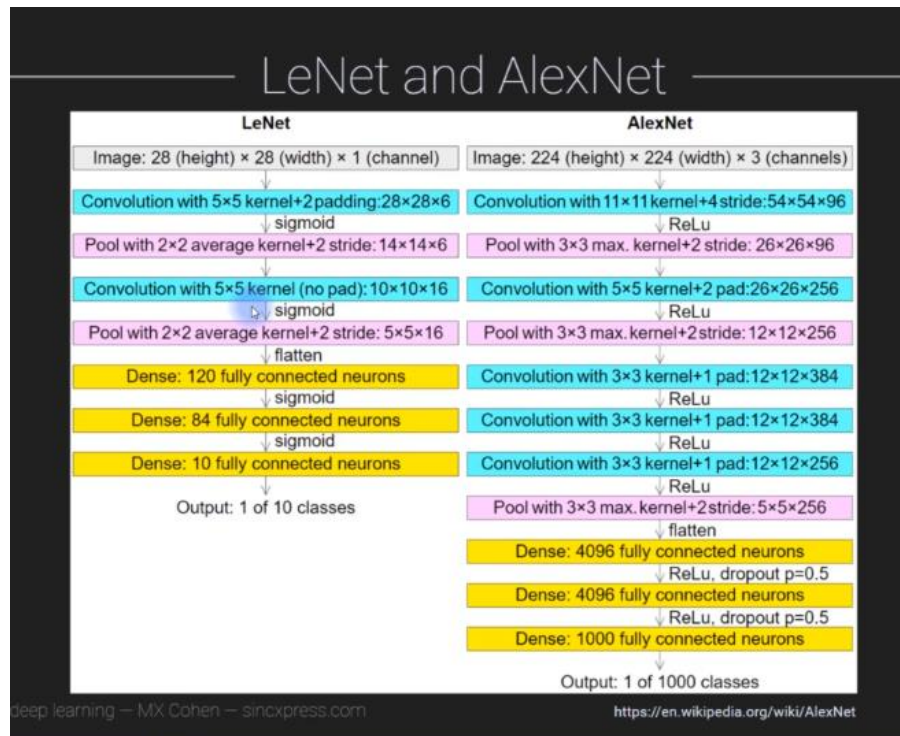
What is Transfer Learning? Transfer Learning is a machine learning method where we reuse a pre-trained model as the starting point for a model on a new task. To put it simply—a model trained on one task is repurposed on a second, related task as an optimization that allows rapid progress when modeling the second task.

LeNet

There are many famous deep learning models that are pre trained and available for you, I am going to give an overview of a few of the more commonly used models. One of these models is the LeNet model. This is the model that really kicked off the CNN design and therefore, it's a really important and influential model in the history of deep learning. It was developed in the late 1990s by a researcher named Lukan and his colleagues.



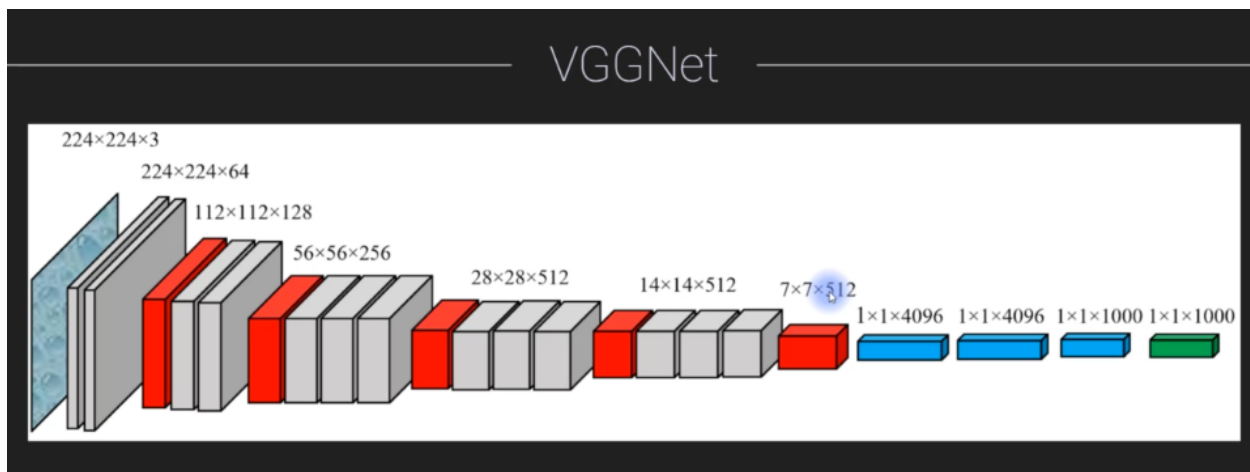
You can see that this architecture has two standard convolution pooling blocks. And we have a couple of feed forward layers which here they call dense layers. In fact, this model is the reason why we call this a standard convolution pooling block. This is the model that made this architecture standard. On the other hand, this model is not terribly deep, nor it is very difficult



The typical CNN architecture has a convolution layer and the immediately thereafter a pooling block layer like this but here see that this is an example where we have multiple coevolution layers stacked on top of each other without any pooling between them.

VGGNET

VGGNET is another famous CNN model which often used in transfer learning.



The VGG is for Visual Geometry Group. This is the name of the research group at oxford university that develop this model. so these gray boxes here represents coevolution layers and the red boxes represent pooling layers. so again you see there are multiple coevolution layers stacked on the top of each other before getting to a max pooling layer and there for allow

image down sampling. we have here two convolution layers and then max pool for down sampling and then two more convolution blocks and the convolution blocks and so on. You see the typical style of CNN architectures that as we go deeper into the model, the image resolution is decreasing, whereas the abstraction of the number of layers, the number of feature maps is increasing. so the image resolution is going from two hundred and twenty four down to one, twelve and fifty six and all the way down to seven by seven. so the image resolution is getting really small here compared to the original image size, but the depth of the representation, so the number of feature layers is increasing from 64 up to 512. so there is actually several different versions of the VGG network. one of this is configuration C which is often called VGG16, because there are 16 layers in this model, including the fully connected layers here at the bottom.

ResNet

ResNet is really long network. The architecture of Resnet is really interesting and powerful complexity to the flexibility of these models. There are many pretrained model that you can find, download and modify to use.

Once you learn how to implement transfer learning, then using a different model is pretty straightforward.

How do I use ResNet transfer learning?

implementing the Resnet-50 model to solve an image classification problem:

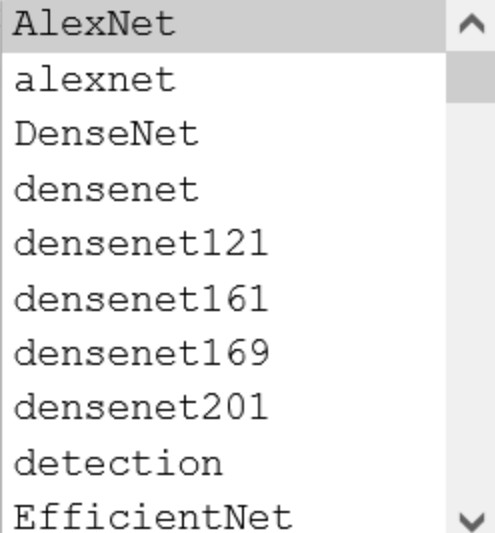
1. Step 1: Import all the required libraries. ...
2. Step 2: Import your dataset. ...
3. Step 3: Split Your Data. ...
4. Step 4: Visualize your data. ...
5. Step 5: Import your Pre-trained Model. ...
6. Step 6: Model Evaluation.

[Reference: <https://chroniclesofai.com/transfer-learning-with-keras-resnet-50/>]

In the torch vision there are several pretrained models as below code you can see the list of the and the next step is to select the model that is near to your purpose and dataset, download it and tune it for your purpose.

```
1 import torch
2 import torchvision
```

```
1 torchvision.models.|
```



AlexNet
alexnet
DenseNet
densenet
densenet121
densenet161
densenet169
densenet201
detection
EfficientNet

You can download only your model architecture without any pre-trained weight if determine the pretrained = False, for example in the below picture we say pretrained = True so we can use the pre trained weight of ResNet model.

```
resnet = torchvision.models.resnet18(pretrained=True)
```

You should pay attention that there are several hyperparameters that we can change in pre-trained model to tune our new model for example in some case maybe using Adam optimizer cause the better result that SGD optimizer. That will be very interesting experiment to explore.

Other good source for more information about transfer learning:

<https://learnopencv.com/pytorch-for-beginners-image-classification-using-pre-trained-models/>

<https://learnopencv.com/image-classification-using-transfer-learning-in-pytorch/>

<https://pytorch.org/vision/stable/models.html>

https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html