Deep learning for computer vision

2021.6.27

5.3 Using a pretrained convnet

A common and highly effective approach to deep learning on small image dataset is to use a pretrained network. A *pretrained network* is a saved network that was previously trained on a large data set, typically on a large-scale image-classification. If this original dataset is large enough and general enough, then the spatial hierarchy of features learned by the pretrained network can effectively act as a generic model of the visual world, and hence its feature can prove useful for many different computer vision problems, even though these newproblems may involve completely different classes than those of the orginial task.

In this case, let's consider a large convnet trained on the ImageNet dataset (1.4 million labeled images and 1000 different classes). ImageNet contains mant animal classes, including different species of cats and dogs. We'll use the VGG16 architecture here.

There are two ways to use a pretrained network: *feature extraction* and *fine-tuning*.

5.3.1 Feature extraction:

Feature extraction consists of using the *representations* learned by a **previous network** to extract interesting *features* from **new samples**. These features are then run through a new classifier, which is trained from scratch.

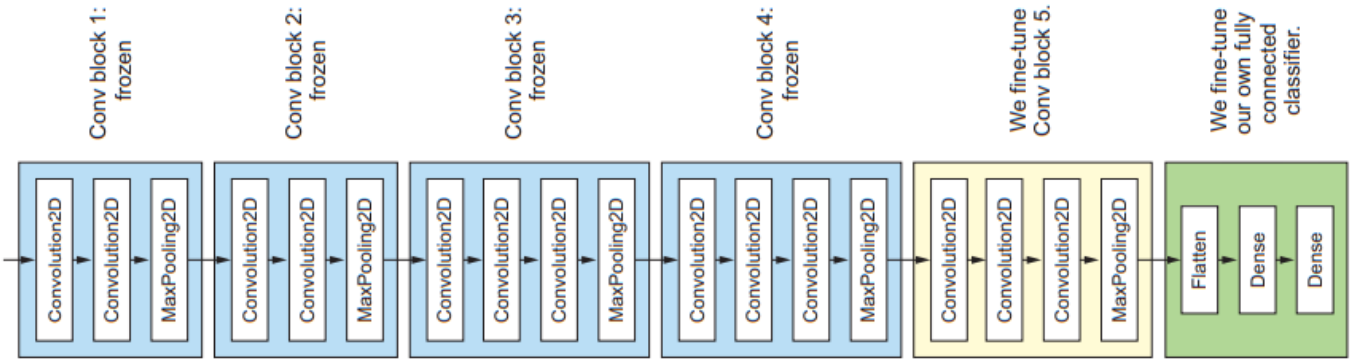
Convnets used for image classification comprise two parts: starting with a series of pooling and convolution layers, and they end with a densely connected classifier. The first part is called the *convolutional base* of the model. In the case of convnets, feature extraction consist of taking the convolutoinal base of a previously trained network, running the new data throught it, and training a new classifier on top of the output.

The representation learned by the convolutional base are likely to be more generic and therefore more reusable: the feature maps of a convnet are presence maps of generic concepts over a picture, which is likely to be useful regardless of the computer-vision problem at hand. But the representations learned by classifier will necessarily be specific to the set of classes on which the model was trained. Additionally, representations found in densely connected layers no longer contain any information about *where* objects are located in the input image: these layers get rid of the notion of space, whereas the object location is still described by convolutional feature maps. For problems where object location matters, densely connected features are largely useless.

Note that the level of generality and reusability of the representations extracted by specific convolution layers depends on the depth of the layer in the model. Layers that come **earlier** in the model extract **local, highly generic** feature maps (such as visual edges, colors, and textures), whereas layers that are higher up extract more-abstract concepts (such as "cat ear" or "dog eye"). So if your new dataset differs a lot from the dataset on which the original model was trained, you may be better off using only the first few layers of the model to do feature extraction, rather than using the entire convolutional base.

5.3.2 Fine-tuning:

Fine-tuning consists of unfreezing a few of the top layers of a frozen model base used for feature extraction, and jointly training both newly added part of the model and these top layers.



Steps for fine-tuning a network are as follows:

1. Add custom network on top of an already-trained base network.
2. Freeze the base network.
3. Train the part you added.
4. Unfreeze some layers in the base network.
5. Jointly train both these layers and the part you added.