VI. Lab Six

Data are one of the most precious resources for developing deep learning projects. In the medical field, a datum may correspond to a very painful patient or a clinical test. In transportation analysis, a datum may correspond to a traffic accident. Hence, transfer learning is a priceless tool in solving the problems brought by insufficient training data for the target model. And it has brought significant positive influence to the areas which were troubled by that problem. Moreover, based on the intuitive thought that knowing how to play the saxophone may help on learning how to play the trumpet, or even on singing and playing the piano, it conjectures that things may can be the same on deep learning.

Accordingly, (deep) transfer learning can be defined as: based on a (deep) learning task A, and it can get help from another (deep) learning task B. It aims to improve the prediction performance of A by discovering and transferring the latent knowledge from B. Besides, for most situations, the data scale of task B is much larger than task A.

In this work, we focus on the classification problem of a dataset that contains 4774 images for training and 1969 for testing, which is from 24 different categories of boats navigating in the City of Venice (Italy). The average amount of each item is reasonable, whereas some are extremely short of samples. Therefore, we aim to remedy this weakness by using transfer learning technologies.

A. Finetuning

I train a typical CNN model with two convolution layers and three fc layers using the torchbearer for 100 epochs, and the results on test dataset are shown below:

```
                    precision recall f1-score support
                Alilaguna 0.90 0.83 0.87 101
                Ambulanza 0.93 0.84 0.88 76
                 Barchino 0.91 0.70 0.79 100
                  Gondola 1.00 0.95 0.98 21
            Lanciafino10m 1.00 0.63 0.77 19
                Motobarca 0.83 0.40 0.54 193
Motopontonerettangolare 1.00 1.00 1.00 16
             MotoscafoACTV 1.00 0.89 0.94 9
                  Mototopo 0.77 0.94 0.85 790
                Patanella 0.75 0.74 0.74 251
                  Polizia 0.86 0.59 0.70 63
           Raccoltarifiuti 0.93 0.75 0.83 84
             Sandoloaremi 1.00 0.91 0.95 11
                    Topa 0.89 0.46 0.60 70
            VaporettoACTV 1.00 1.00 1.00 854
                   Water 0.94 0.99 0.97 816
```

The training loss curve of it will be shown later, which can denote its low convergence speed. And subsequently, I import a pre-trained ResNet50 of the ImageNet from the 'torchvision.models' and explore the performances between different fine-tuning approaches. There are three main ways to fine-tune a transferred model, namely are:

1) To remove the last few layers (fc layers) of the pre-trained network and freeze the weights of the remaining layers (e.g. stop monitoring their weights) in the pre-trained network, and further train the new-added layers with our own data and labels.
2) To remove the last few layers (fc layers) of the pre-trained network and connect it with other machine learning approaches that correspond to our goal task. In other words, we keep using the original feature extracting function of the transferred model, and then achieve our goal task by focusing on the correlations between output features and our own model.
3) Use a smaller learning rate to further train the whole network with our own data and labels.

Firstly, I use the first approach based on the pre-trained resnet50 model of the ImageNet from 'torchvision.models'. We train it with the same 100 epochs, and it achieves a perfect performance but uses considerable time. And we can tell it has better convergence speed compared with the typical CNN model above which is shown in the graph below.

```
                    precision recall f1-score support
                Alilaguna 0.98 1.00 1.00 101
                Ambulanza 1.00 1.00 1.00 76
                 Barchino 0.99 1.00 1.00 100
                  Gondola 1.00 1.00 1.00 21
            Lanciafino10m 1.00 0.95 0.97 19
                Motobarca 0.97 0.99 0.98 193
Motopontonerettangolare 1.00 1.00 1.00 16
             MotoscafoACTV 1.00 1.00 1.00 9
                  Mototopo 1.00 0.99 1.00 790
                Patanella 0.99 0.96 0.98 251
                  Polizia 1.00 1.00 1.00 63
           Raccoltarifiuti 1.00 1.00 1.00 84
             Sandoloaremi 1.00 1.00 1.00 11
                    Topa 0.92 0.99 0.95 70
            VaporettoACTV 1.00 1.00 1.00 854
                   Water 1.00 1.00 1.00 816
```

B. Reflect on the two Different Approaches.

For the second approach, in the beginning, I measured the Euclidean distance of the features which come from the original feature extracting function, between several random pairs of samples with the same label, as well as it between several random pairs of samples with the different label. It is a common phenomenon that the distance between samples in the same cluster is much smaller than in the different groups.

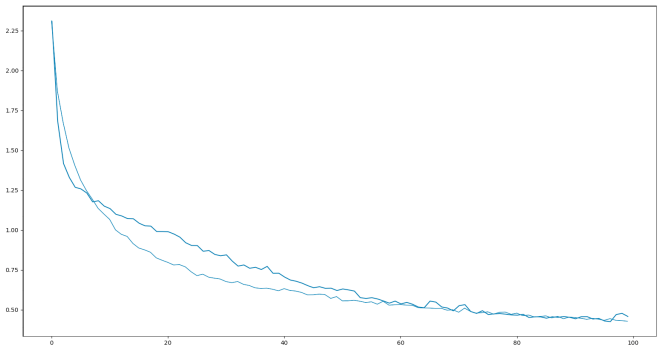

Figure 13: Loss curves of both models above, the lower one is the resnet50-based model while the upper one is the typical CNN.

We connect that part with the support vector classification in 'sklearn', and, amazingly, the inference processes of training, validation and testing only took me around twenty seconds without using GPUs. The prediction accuracy is worse than the typical CNN, but still acceptable.

	precision	recall	f1-score	support
Alilaguna	0.90	1.00	0.95	19
Ambulanza	0.82	0.82	0.82	22
Barchino	0.76	0.25	0.38	51
Gondola	1.00	0.67	0.80	3
Lanciafino10m	0.00	0.00	0.00	7
Motobarca	0.82	0.31	0.44	59
Motopontonerettangolare	1.00	1.00	1.00	3
MotoscafoACTV	0.00	0.00	0.00	1
Mototopo	0.82	0.99	0.89	274
Patanella	0.41	0.84	0.55	74
Polizia	0.67	0.13	0.22	15
Raccoltarifiuti	1.00	0.74	0.85	19
Sandoloaremi	0.00	0.00	0.00	3
Topa	0.00	0.00	0.00	29
VaporettoACTV	0.99	1.00	1.00	325
Water	0.99	0.97	0.98	420

For the third approach, it takes years to train the whole resnet50 by our dataset for 100 epochs with a small learning rate. Also, under the limitation of the computational ability of my device, it isn't easy to draw a conclusion about the influence of adjusting the learning rate on the performance of a deep neural network. However, I explored the same things in the section above, in a simple neural network. By way of conclusion, first approach is a good point between the computational costs and the performance; second approach takes the biggest time costs and the performance also depends; the third approach of fine-tuning is the fastest one, and its performance is still acceptable.