IN3310

Week 05: Fine Tuning of neural

networks

1 two exemplary ways of fine-tuning, setup with a separate validation set

- take the 102 class flowers dataset and write a dataset class which can work with a train/val/test split for it. The difference to the 102 flowers from oxford is that the train/val/test split is provided for your convenience. You can find the flowers dataset on the course page, under exercises.

- take any deep network you like which has pretrained weights (a smaller resnet, a smaller densenet are training fast)
- train a deep neural network in three different modes
  - (A) once without loading weights and training all layers.
  - (B) once with loading model weights before training and training all layers,
  - (C) once with loading model weights before training and training only the last **one** trainable layer (note: for quite some problems, the approach B is better than C)

  For each of these 2 modes select the best epoch by the performance of the model on the validation set. Typically less than 20 epochs should suffice for training when using finetuning. You can run also optionally a selection over a few learning rates, if you use a GPU.

What do you need to do for steps when you start with a code like the MNIST training code?

- write a new dataloader for your training dataset, the flowers
- adjust paths for data (and if necessary for label paths/files or files determining splits into train/val/test)
- write the code so that it works with paths relative to the directory of your script
- as you did not have data augmentation so far (next lecture), you can simply use the following:

```
data_transforms = {
    'train': transforms.Compose([
        transforms.Resize(256),
        transforms.RandomCrop(224),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
    ]),
    'val': transforms.Compose([
        transforms.Resize(224),
        transforms.CenterCrop(224),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
    ]),
}
```

We are aware that we are using the imagenet mean and the imagenet standard deviation.

- use some deep learning model from the model zoo, load its weights before training for settings B and C
- adjust the number of classes in the last linear/dense layer
- in the optimizer provide the proper parameters for training for settings A,B,C
- collect and plot curves of the training loss, the validation set loss and the validation set accuracy. also print the final test accuracy of the selected model
- A note: Calling a model constructor with `pretrained=True` does not tell you what really goes on. Check `https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py` to see what routine is used to load a model.

2 Theory Questions 1-3:

- You are given a 2-dimensional convolution with feature map input size $(78, 84)$. When using a kernel of size $(5, 5)$ and stride 3 with padding of 2, what will be the spatial size of the feature map which is the output of the convolution? Note that the *spatial* size does not depend on the number of input or output channels.
- You are given a 1-dimensional convolution. When using a kernel of size 9 and stride 3 with padding 1, which spatial input size do you need to have, so that you have a spatial output size of 16?
- You are given a 2-dimensional convolution. When using a kernel of size $(3, 5)$ and stride 2 with padding of 0, what will be the spatial size of the feature map which is the output of the convolution, which spatial input size do you need to have, so that you have a spatial output size of $(128, 96)$?

3 Theory Questions 4-6:

How many trainable parameters are in

- a 2-D convolutional layer with input $(32, 19, 19)$, kernel size $(7, 7)$, stride 3, 64 output channels?
- a 2-D convolutional layer with input $(512, 25, 25)$, kernel size $(1, 1)$, stride 1, 128 output channels?
- how many multiplications and how many additions are performed in the first case above?