Image Classification Project

- This assignment is the machine learning project for classifying images. You will have to train a machine learning model to predict the 20 possible classes of images. The training set contains these labels (along with the number of images in each class):

Class ID	Class Name	Count
0	book	739
1	bottle	715
2	car	500
3	cat	564
4	chair	508
5	computermouse	534
6	cup	774
7	dog	464
8	flower	875
9	fork	605
10	glass	572
11	glasses	436
12	headphones	459
13	knife	790
14	laptop	446
15	pen	954
16	plate	433
17	shoes	899
18	spoon	625
19	tree	591

A subset of the images collected (12483 images) is provided for training.

Note that the images have been validated and thus have numbered file names, where the labels are stored separately in the accompanying `labels.csv` file. Furthermore, the images have been resized to 100 pixels (either width or height, depending on the aspect ratio).

Trained Model

Solution contains:

- architecture.py: this script contains your model class and the instantiated object. Note that the name has to be exactly architecture.py, because this is expected by the evaluation script on the server. Also, the class name has to be MyCNN, the variable name containing the instance has to be model. Example:

```
import torch
import torch.nn as nn

class MyCNN(nn.Module):
    def _init_(<your params>):
        super()._init_()
        <your code>

    def forward(self, input_images: torch.Tensor) -> torch.Tensor:
        <your code>

model = MyCNN(<your params>)
```

- model.pth: this file contains your trained model. Again, use this exact same file name. To create this file, use torch.save(model.state_dict(), "model.pth"). So in your training loop you need to store the trained model using this code (preferably the one with the lowest loss). You can load the file again with model.load_state_dict(torch.load(trained_model)). On the website you can see the code of the evaluation script, so you can make sure that your submission is compatible.

The training set images have been reduced to a size of either 100 × _ or _ × 100 pixels, depending on the format of the image. For the evaluation on the hidden test set on the challenge server, note that the images need to have a dimensionality of 100 × 100 pixels. To create the data set, class ImageDataset, along with the functions to_grayscale and prepare_image were used. Import them using the file dataset.py - note that ImageDataset is slightly different from the version in assignment 3 (normalization and tensor conversion was added), so use this one to ensure compatibility with how the test set is generated.

The solutions predicts the labels for the images of the test set.