Image Classification Project

This assignment is the machine learning project for classifying images. There are two submissions that must both be uploaded until the specified due date:

- a ZIP file of the model architecture and the trained model to the dedicated challenge server (username = password = `K<8-digit-matr-ID>`)
- the project files to Moodle

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 in Assignment 1, Exercise 1 (12483 images) is provided for training (download), the remaining images serve as hidden test set.

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).

Exercise 1 – Submission: Trained Model (Challenge Server) 200 Points

Your submission for the challenge server has to be a ZIP (*only* zip - no tar, 7z, etc.) archive that contains two files:

- `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 $100 \times \_$ or $\_ \times 100$ 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.

Your task is to predict the labels for the images of the test set. The evaluation criterion is overall accuracy on the test set.

The following restrictions apply:

- You only have 10 valid attempts to upload a model.
- Invalid attempts (e.g., error parsing your submission files) are not counted.
- Your best attempt will be used as final attempt.

The points are determined as follows:

- You will get the full number of points if you achieve at least 50% accuracy on the test set.
- If you have less than 50% accuracy, but more than 30%, you will get points increasing linearly from 10 (31%) to 200 (50%). 30% accuracy or less will result in 0 points. For points calculation,

percentages will be rounded to the nearest full percent.

- For every percentage point more, you get 1 bonus point. So for example, if your accuracy on the test set is 65% you get 215 (200 + 15) points.

See the leaderboard for the individual accuracies. When you upload your submission, make sure that you always enter your full name.

Exercise 2 – Submission: ZIP Archive With Project Files (Moodle) 200 Points

In addition to the challenge server submission, you must also upload all of your project files to Moodle. This will be used to check for plagiarism and to verify that your model corresponds to the uploaded submissions.

Create a ZIP archive that includes *all* project files, i.e., the entire source code, all configuration files and all documentation files. The filename of the ZIP archive can be arbitrary. You do not have to include the training data itself, however any scripts that augment the training data need to of course be included so that we can exactly replicate your result.

General Project Hints:

- Divide the project into subtasks and check your program regularly.
  - Decide which samples you want to use in your training, validation or test sets.
  - Check the units on data loading, neural network inference and training and the respective code files and decide which elements are useful for your implementation.
  - Implement the computation of the loss between NN output and target (see code files of Unit 7).
  - Implement the NN training loop (see code files of Unit 7),
  - Implement the evaluation of the model performance on a validation set (see code files of Unit 7).
- You can have a look at the example project in the code files of Unit 7. This is not a classification project, but you can still check whether you can use parts of the code as well.
- Set a random number seed in order to obtain reproducible results.
- Do not use transfer learning, you should create a model from scratch. You can of course research published models, but do not copy 1:1.
- You only have 10 attempts to submit predictions, so it will be important for you to use some samples for a validation set and maybe another test set to get an estimate for the generalization of your model.
- You do not need to reinvent the wheel. Most of this project can be solved by reusing parts of the code materials and assignments from this semester.
- When uploading to the challenge server, you might not immediately see a result due to the server's task scheduling. Please be patient and check back later. This also means that you should try to upload your model in due time to avoid having no immediate feedback when the assignment deadline approaches.