Introduction to Python and PyTorch

Prof. Anna Rohrbach,
Prof. Marcus Rohrbach,
Marcel Klemt

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Summer Semester 2024 – Multimodal Artificial Intelligence
Deadline Wednesday 8th May, 2024
Sheet 1

General Information

Exercises will not be graded. However, they prepare you for the project and the final exam.

- The first few exercises serve as a more technical/coding introduction to prepare you for the project.
- The later exercises will prepare you for the final exam.

For each exercise sheet, we will have an exercise session where you can ask questions and discuss solutions. Feel free to ask or even answer questions to/from your peers in the Content Q&A MoodleOverflow. Question posts will be anonymous.

You can find the exact dates of the exercise sessions in Moodle. We're going to use some of the session slots for the projects, so not every session will be an exercise session.

Solutions of the exercise sheet will be published after the corresponding exercise session on Moodle and GitHub.

Task 1.1: Set up Google Colab

In this course, we use Google Colab to access GPUs. To use Google Colab you need to log in with a Google account and open a notebook. Here, you can create a new notebook or open an existing one by providing its URL.

To import the provided notebook, click on "GitHub" and insert `https://github.com/multimodal-ai-lab/multimodal-ai-course` as the GitHub URL, see Figure 1. From the "Paths" list, select the entry corresponding to exercise 1.

If that doesn't work, you can still download the notebook from GitHub and re-upload it to Colab.

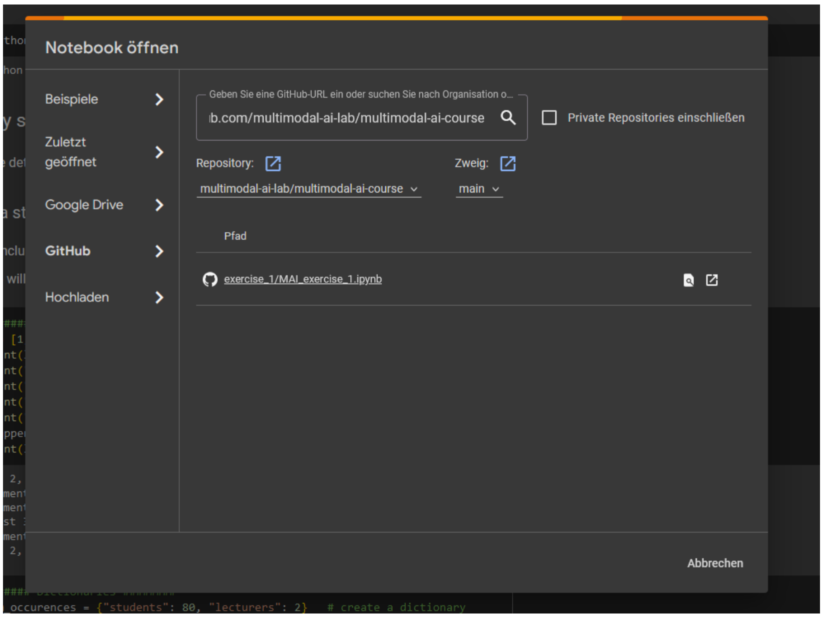
Caution: In order to save your changes (*ctl* + *s*) you will be asked to save a copy of the notebook in your Google Drive. Do that! Otherwise, your changes won't be saved, and the notebook will reset.

The notebook should open. Only a single CPU is assigned—this is sufficient for our Python, NumPy, and Matplotlib introduction. However, a GPU is advantageous for the later PyTorch introduction, as it considerably shortens compute time.

You can change the runtime type (switch between CPU and GPU) in the top-right menu (Figure 2). Be careful: This will reset the execution of your notebook, i.e., all information stored in variables will be lost, and you will have to execute all cells again.

Task 1.2: Introduction to Python

The notebook starts with a short introduction to Python and the two important libraries: NumPy and Matplotlib.


Notebook öffnen
Beispiele
Zuletzt geöffnet
Google Drive
GitHub
Hochladen
Geben Sie eine GitHub-URL ein oder suchen Sie nach Organisation o...
□ Private Repositories einschließen
Abbrechen

ib.com/multimodal-ai-lab/multimodal-ai-course

Repository: multimodal-ai-lab/multimodal-ai-course

Zweig: main

Pfad

exercise_1/MAI_exercise_1.ipynb

Figure 1: Notebook opening view.

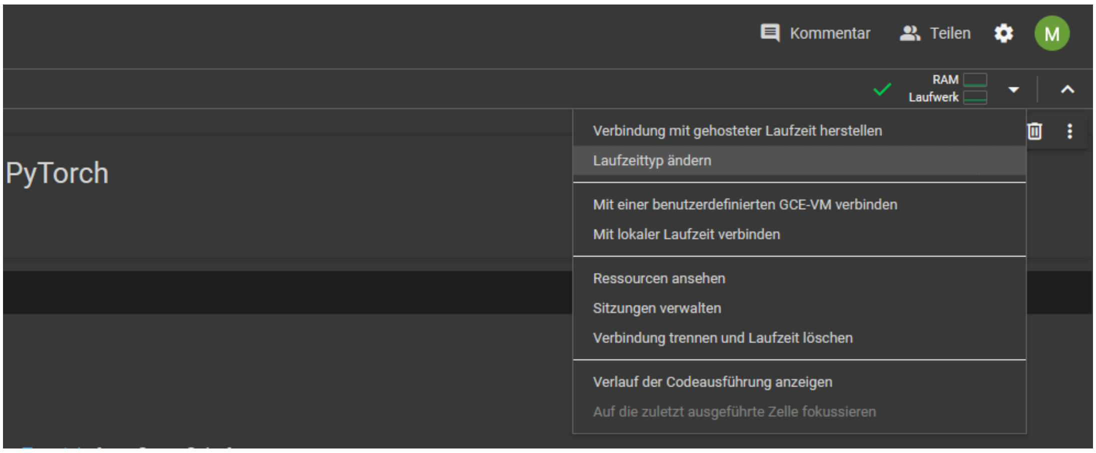

Kommentar
Teilen
M
RAM
Laufwerk
PyTorch
Verbindung mit gehosteter Laufzeit herstellen
Laufzeittyp ändern
Mit einer benutzerdefinierten GCE-VM verbinden
Mit lokaler Laufzeit verbinden
Ressourcen ansehen
Sitzungen verwalten
Verbindung trennen und Laufzeit löschen
Verlauf der Codeausführung anzeigen
Auf die zuletzt ausgeführte Zelle fokussieren

Figure 2: Where to change the runtime type.

1.2a) Basics

Please read through the intro (down until ”NumPy”) and, if you like, play around with the examples. This is a Jupyter Notebook, so you can arbitrarily modify and execute the code cells.

1.2b) Try out NumPy

In the NumPy part of the notebook, we included a small exercise where you implement the **cosine similarity** using only the NumPy library. The cosine similarity is a metric to measure the similarity of two vectors v_1 and v_2. It is defined as

$$\cos(\theta) = \frac{v_1 \cdot v_2}{||v_1|| \cdot ||v_2||}, \tag{1}$$

where θ is the angle between those vectors.

The subsequent cell is a test cell to verify your implementation. Just run it, it works out of the box.

1.2c) Try out Matplotlib

The Matplotlib part of the notebook includes a small exercise where you visualize the sine and cosine waves as a line plot and adjust the x-axis. As this is difficult to test automatically, we will discuss it during the exercise session.

1.2d) Optional: Additional learning resources

A complete introduction would go beyond the scope of this exercise (and this course). For further resources, we can recommend the Python Beginner Tutorials from Corey Schafer. He also has a playlist about Matplotlib, which is a library for creating visualizations like diagrams, charts, etc. Sadly, he doesn't have a video about NumPy, but here we can recommend the NumPy for ML tutorials from Codemy.com.

Another powerful library is Pandas. We won't cover it in this exercise but, nevertheless, it is also useful for ML. Pandas allows fast and easy handling of data structures, which might be helpful for your projects. Luckily, Corey Schafer provides Pandas Tutorials as well.

Task 1.3: Introduction into PyTorch

This part provides you a gentle hands-on experience with PyTorch.

1.3a) Train your own image classifier

Train a small classifier on CIFAR10, visualize some images with the corresponding labels, and take a look at the training progress.

1.3b) Plot the loss with Matplotlib

Plot the list of losses gathered during training as a line chart to see whether the loss decreases.

1.3c) Test your model

We provided an image of a deer and a cute cat (also in the GitHub repo). (You need to upload the image to your Colab instance or your Google Drive and make it accessible to your Colab instance.) Predict the label of the images with your trained model. Let's see if your model gets it right.

1.3d) Improve your model

The provided setup is not perfect. The model learns something, as it achieves a higher accuracy than a random prediction, which would be 10%; however, there is much room for improvement. What are possible ways to improve the performance? Feel free to implement them.

1.3e) Optional: More learning resources

To learn more, check out the PyTorch Learn the Basics tutorial or, if you prefer, its video version Deep Learning with PyTorch.

More information about individual PyTorch layers needed to build your model can be found on the official website.