Universität Bremen

Cognitive Systems Lab

Advanced Machine Learning Tutorial

Getting Started with PyTorch

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul,
and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

Outline

- → Organization
- → What is PyTorch?
- → PyTorch in practice
  - Example machine learning task
  - Data
  - Dataset/Dataloader
  - Model
  - Loss/Objective function
  - Optimizer
  - Training, validation, and testing cycles
  - Save/load model and optimizer

Basic components

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

Organization

- If you haven't done so, please assign yourself to one of the tutorial slots on Stud.IP via Participants/Teilnehmer →Groups/Gruppen
- **Portfolio:**
  - determines grade for this course
  - exercises will be handed out as a jupyter notebook template after the last lecture
  - work in groups of 3 for **one week**
- **Grade Bonus:**
  - improves grade by up to **two grade steps** (0.7) if portfolio is passed
  - Two bonus paths:
    - **First grade step**: Present your solution to one of the exercises from the 5 exercise sheets **AND** show active participation throughout the tutorial sessions (alternatives if not possible)
    - **Second grade step**: Work on a mini project and present the result in the tutorial sessions (08./09.07 + 15./16.07) (topics released on 22./23.04)

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

Organization

- **Portfolio Groups:**
  - If you already have a group: please tell us who is in your group. If one member drops out, find yourself a substitute.
  - If you are looking for a group or a member for your group: please ask in Stud.IP’s forum or stay here after the tutorial.
  - Try to find someone in your tutorial if possible

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

Organization

- **Exercise sheets:**
  - You don’t need to hand them in and they are not graded but presenting the solution to one exercise is required for receiving the grade bonus.

Topic	Release	Presentation in Tutorial
1. Fundamentals, Intro NN	24.04	06.05/07.05
2. End-to-End, RNN	08.05	??.05/21.05
3. CNN, Attention	22.05	03.06/04.06
4. Generative Models, Transformer	05.06	17.06/18.06
5. Distance Metric Learning, Explainable AI	19.06	01.07/02.07

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

Organization

- **Mini-Project:**
  - chance for more practical work on a bigger project than the exercise sheets
    - Example: Implement a variant of a neural network architecture from a paper and evaluate it against the baseline model from the lecture
  - We will propose topics to choose from, you can also propose your own ones (with our approval)
  - Timeline:
    - Topic presentation: 22./23.04
    - Choosing topics: Until 15.05.
    - Presentations of mini projects in the tutorial sessions (08./09.07 + 15./16.07)

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

What is PyTorch?

PyTorch is a Python framework which allows to develop machine learning solutions for diverse problems. It provides a comprehensive collection of different tools beginning with simple mathematical operations and ending up with data parallelized deep neural network training. PyTorch can be used together with the standard Python packages.

Further information: https://pytorch.org/

PyTorch documentation: https://pytorch.org/docs/stable/index.html

PyTorch github page: https://github.com/pytorch/pytorch

PyTorch tutorials: https://pytorch.org/tutorials/

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

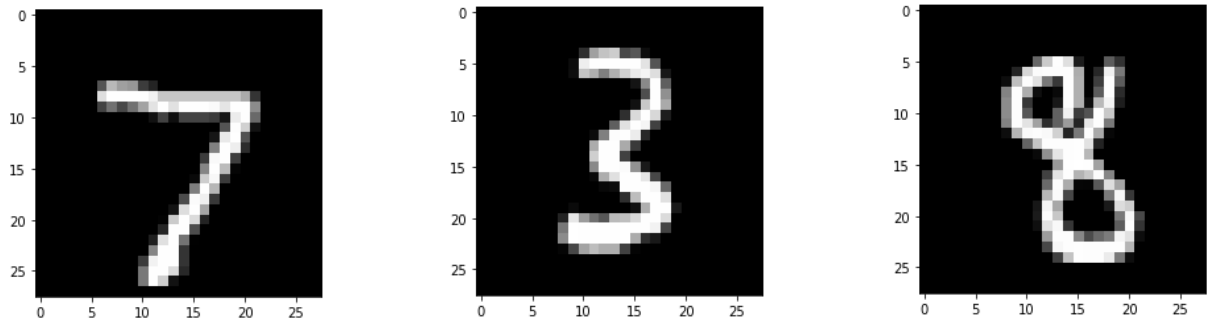
PyTorch in practice

Example machine learning task

A machine learning model, more precisely a neural network, shall be used to classify images of handwritten digits (0-9). Each image belongs to one out of ten classes.

We will use the **M**odified **N**ational **I**nstitute of **S**tandards and **T**echnology (MNIST) database
http://yann.lecun.com/exdb/mnist/

Each image is given as a 28x28 pixels gray scale value image. In total, the database comprises 50.000 and 10.000 images for developement and testing respectively.



Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Data

The data is used to train, validate, and test your machine learning model.

Usually, the complexity of a task defines the required amount of data.

Depending on the task, the data can comprise one or several features.
Question: Which database examples for multi-feature data do you know? For what ML problems are they suitable?

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Data

The data is used to train, validate, and test your machine learning model.

Usually, the complexity of a task defines the required amount of data.

Depending on the task, the data can comprise one or several features.
Question: Which database examples for multi-feature data do you know? For what ML problems are they suitable?

- Local information of urban quarters (distance to next hospital, crime rate, etc.) -> ?
- Weather data -> ?
- Human activity recordings (sensor data) -> ?
- Audio-visual speech data (videos) -> ?

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Dataset/Dataloader

Usually huge amount of data -> Cannot fit into memory at once.

Solution: Dataset and dataloader.

The dataset class defines how to lazy load only the required data into the memory. You have to implement this according to the data/task.

The dataloader interfaces the dataset and takes care of training/validation/test set details.

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Dataset/Dataloader

```
class myDataset(Dataset):
    def __init__(self, data_list):
        self.data_list = data_list          # List contains data paths

    def __len__(self):                      # Get total number of dataset samples
        return len(self.data)

    def __getitem__(self, idx):             # Prepare an item
        current_path = self.data_list[idx]  # Get current path
        input = ...
        label = ...
        return _input, label                # Return data and label

my_dataset = myDataset(filepath)
my_dataloader = DataLoader(my_dataset, batch_size=64, shuffle=True) # Wrap into PyTorch dataloader
```

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Model

The core of the machine learning system.

Receives the input data, performs some processing, and predicts the output data.

Any kind of model architecture is possible.
Examples:

- Simple linear regression
- Advanced image classification
- Complex acoustic modeling

In general: There is no one optimal architecture (further reading: “No-free-Lunch-Theoreme“).

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Model

```
# Define model (neural network)
class myNetwork(nn.Module):
    def __init__(self):
        super(myNetwork, self).__init__()
        self.conv1 = nn.Conv2d(1, 10, kernel_size=5)
        self.fc1 = nn.Linear(1440, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = F.max_pool2d(x, 2)
        x = F.relu(x)
        #x = x.view(-1, 1440)
        x = th.flatten(x, start_dim=1)
        x = self.fc1(x)
        return x
```

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Loss/Objective function

A loss/objective function is based on a metric (or a combination of several ones) to determine and assess the difference between the model‘s output and the ground truth data.

Needs to be differentiable to allow the calculation of gradients.

There is no universal loss since it has to match the problem.
Examples:

- Classification of two classes: *Possible loss?*
- Classification of ten classes: *Possible loss?*
- Estimation of a sound signal: *Possible loss?*

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Loss/Objective function

A loss/objective function is based on a metric (or a combination of several ones) to determine and assess the difference between the model‘s output and the ground truth data.

Needs to be differentiable to allow the calculation of gradients.

There is no universal loss since it has to match the problem.
Examples:

- Classification of two classes: Binary cross entropy loss
- Classification of ten classes: Categorical cross entropy loss
- Estimation of a sound signal: Mean squared error or signal to noise ratio loss

A training/validation loss also depends on the data representation and can be different from the final evaluation metric.

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Loss/Objective function

Cross entropy loss out of the box, provided by PyTorch:

```
th.nn.CrossEntropyLoss() # Loss out of the box -> Applies log_softmax(), followed by negative log likelihood loss
```

Mean Squared Error loss as custom loss:

```
# Custom loss
def my_mse(true, pred):
    error = true-pred
    squared = error**2
    mean = th.mean(squared)
    return mean
```

$$MSE(y,\tilde{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Optimizer

The optimizer updates the model‘s parameters during the training.

The update calculations are based on the optimization algorithm, different parameters, and the loss function.

Goal: Find the global minimum on the error surface.

Example optimization algorithms:

- Gradient Descent
- Stochastic Gradient Descent (SGD)
- Root Mean Square Propagation (RMS Prop)
- Adam

```
model = myNetwork()
optimizer = optim.Adam(model.parameters(), lr=0.01, weight_decay=0.5)
```

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Training, validation, and testing cycles

Question: Assume you have initialized all required components. What steps do you need to train, validate, and test the model?

Training

Validation/Testing

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Training, validation, and testing cycles

Question: Assume you have initialized all required components. What steps do you need to train, validate, and test the model?

Training

1. Set model to training mode
2. Clear optimizer gradients
3. Feed data to model for prediction
4. Calculate loss
5. Do backpropagation
6. Update model parameters

Validation/Testing

1. Set model to evaluation mode
2. Turn off gradients
3. Feed data to model for prediction
4. Calculate loss

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Save/load model and optimizer

Question: What are we going to do after our model has been trained, validated, and tested?

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Save/load model and optimizer

Question: What are we going to do after our model has been trained, validated, and tested?

Option 1: Discard it.

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Save/load model and optimizer

Question: What are we going to do after our model has been trained, validated, and tested?

Option 1: Discard it.

Option 2 (probably better): Save it!

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Save/load model and optimizer

Question: What are we going to do after our model has been trained, validated, and tested?
Option 1: Discard it.
Option 2 (probably better): Save it!

Save your current model‘s and optimizer‘s parameters (state dictionaries) for future utilization, e.g. for evaluation or to continue the training.

Useful for various applications such as fine-tuning, continuous learning, domain adaptation, transfer learning, multi-task learning, etc.

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Save/load model and optimizer

Question: What are we going to do after our model has been trained, validated, and tested?
Option 1: Discard it.
Option 2 (probably better): Save it!

Save your current model‘s and optimizer‘s parameters (state dictionaries) for future utilization, e.g. for evaluation or to continue the training.

Useful for various applications such as fine-tuning, continuous learning, domain adaptation, transfer learning, multi-task learning, etc.

Are there any possible issues?

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Save/load model and optimizer

Question: What are we going to do after our model has been trained, validated, and tested?
Option 1: Discard it.
Option 2 (probably better): Save it!

Save your current model‘s and optimizer‘s parameters (state dictionaries) for future utilization, e.g. for evaluation or to continue the training.

Useful for various applications such as fine-tuning, continuous learning, domain adaptation, transfer learning, multi-task learning, etc.

Are there any possible issues?
Yes! If the model’s architecture has changed after saving both the parameters and the optimizer, you cannot simply load the model and optimizer anymore. You need to manually solve these issues and e.g. drop parameters, create a new optimizer, etc..
-> A good software version control is always helpful!!

Abdul Haq Azeem Paracha, Anthony Mendil, Daniel Reich, Felix Putze, Kim Marie Lankenau, Rinu Elizabeth Paul, and Prof. Dr.-Ing. Tanja Schultz
Bremen 17.04.2023

Cognitive Systems Lab

PyTorch in practice

Save/load model and optimizer

Furthermore, the entire model can be saved -> Huge storage consumption e.g. when checkpoints are saved.

Further information: https://pytorch.org/tutorials/beginner/saving_loading_models.html

```
th.save(model.state_dict(), os.path.join(save_path, 'model_state_dict.pt'))
th.save(optimizer.state_dict(), os.path.join(save_path, 'optimizer_state_dict.pt'))

model2 = myNetwork()
model2.load_state_dict(th.load(os.path.join(save_path, 'model_state_dict.pt')))
optimizer2 = optim.SGD(model.parameters(), lr=0.01, momentum=0.6)
optimizer2.load_state_dict(th.load(os.path.join(save_path, 'optimizer_state_dict.pt')))
```