DL Exercise 4: PyTorch and Classification Challenge

K. Breininger, V. Christlein, Z. Yang, L. Rist, M. Nau, S. Jaganathan, C. Liu, N. Maul, L. Folle, M. Zinnen, K. Packhäuser
Pattern Recognition Lab, Friedrich-Alexander University of Erlangen-Nürnberg
October 17, 2022

Goal of this exercise

- Get to know a widely used deep learning framework: PyTorch
- Implement & train a variation of a widely used architecture: ResNet
- Classification on **real** data: Images from solar panels

Goal of this exercise

- Get to know a widely used deep learning framework: PyTorch
- Implement & train a variation of a widely used architecture: ResNet
- Classification on **real** data: Images from solar panels
- **Challenge yourself & your colleagues!**

Organizational

Part I: Classification with PyTorch - Mandatory

- Implementation & training of a PyTorch architecture
- Submission of trained models in submission system (later more)
- Code upload to StudOn
- Goal: reach a mean F1 score > 0.60
- **Deadline: TBA**

Organizational

Part II: Challenge - Optional, but highly encouraged

- Try to find & train the best architecture & model for this task!
- Compete with your colleagues!
- **Deadline: TBA**

Source: Designed by Freepik

Data set: Identification of defects in solar panels

- Solar modules are composed of cells
- Are subject to degradation (transport, wind, hail, ...)
- Different defects, e.g., cracks or inactive regions

Data set: Identification of defects in solar panels

- Solar modules are composed of cells
- Are subject to degradation (transport, wind, hail, ...)
- Different defects, e.g., cracks or inactive regions
- **Task:** Automatically determine which defect(s) a module has
- Panel can have no or multiple defects ➔ **multi-label problem**!

Data set: Identification of defects in solar panels

- Solar modules are composed of cells
- Are subject to degradation (transport, wind, hail, ...)
- Different defects, e.g., cracks or inactive regions
- **Task:** Automatically determine which defect(s) a module has
- Panel can have no or multiple defects ➔ **multi-label problem**!

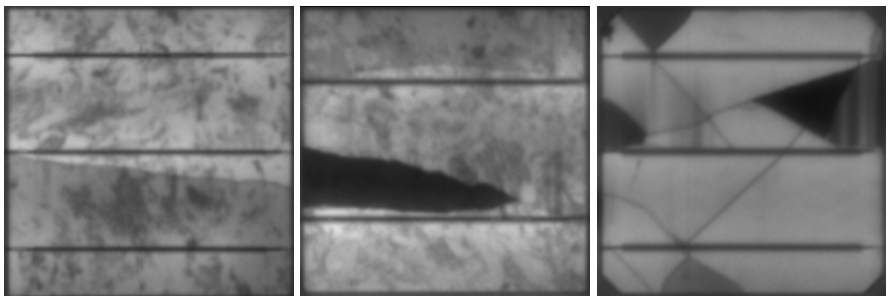


Figure: Left: Crack on a polycrystalline module; Middle: Inactive region; Right: Cracks and inactive regions on a monocrystalline module

Normalization

- The normalization of your implementation has to match the normalization of our test server
- Mean μ and standard deviation σ of the intensity over all test samples are known
- We normalize every pixel x by $x^* = \frac{x-\mu}{\sigma}$
- Please make sure that you implement the normalization accordingly

Deep Learning in PyTorch

We will use **PyTorch** to define and train neural network architectures.

- Developed by Facebook's AI Research lab
  - Open-source
  - Extensive Python interface
- Allows to easily define computational graphs
  - Operations based on **tensors**
  - Closely resembles NumPy API
  - Automatic differentiation to support efficient gradient computations (**Autograd**)
  - Various optimization algorithms to help training neural networks
- + GPU acceleration!

Deep Learning in PyTorch

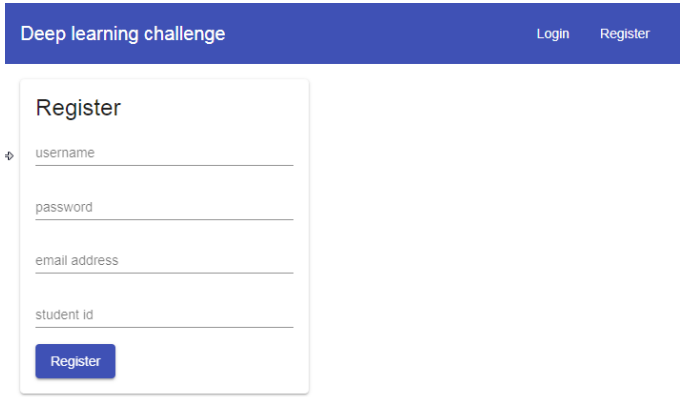
- PyTorch layer API resembles structure of our framework
- Extensive documentation and "getting started" guides
- Short Hands-On will follow after Ex. 3 submission
- Sources online e.g.:
  - 60-min blitz with Jupyter notebooks
  - PyTorch with examples
  - Overview of all tutorials

Submission to online tool

- After training, make sure to save a checkpoint of your best performing model
- Online submission tool will be made available on **TBD**
- Website: http://lme156.informatik.uni-erlangen.de/dl-challenge
- **Only available from within the university network**
- Same teams (max. 2) as before allowed

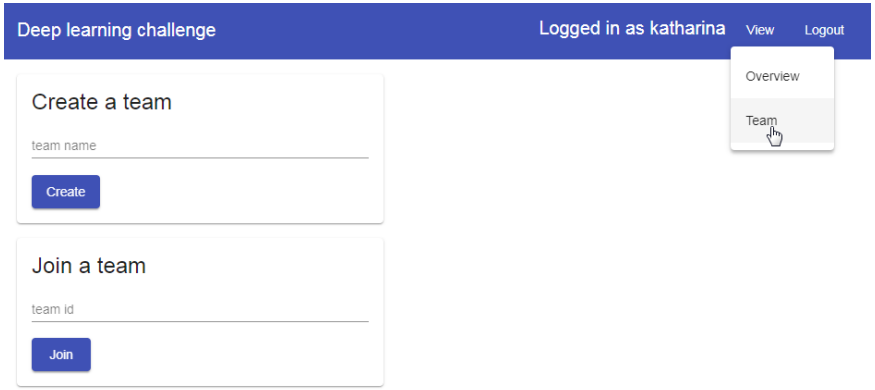
Submission to online tool: Registration

Register with your email and student id.



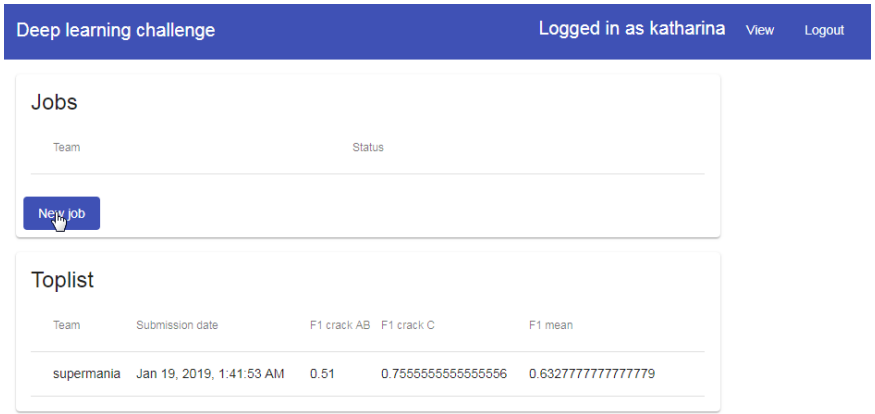
Submission to online tool: Team

If you work in a team: One of you has to create a new team, the other has to join.



Submission to online tool: Submit model

Submit trained models (zip-file generated by train.py) by uploading them. You may submit multiple models.



THE CHALLENGE

Improve on the baseline of ResNet:

- Adapt architecture/try out new architectures
- Pretraining?
- Regularization?
- Data augmentation?
- Use your creativity!
- Best model from each team will be tested on independent data after the challenge deadline
- **Best participants will receive a winner's certificate and a prize!**

THE CHALLENGE

Improve on the baseline of ResNet:

- Adapt architecture/try out new architectures
- Pretraining?
- Regularization?
- Data augmentation?
- Use your creativity!
- Best model from each team will be tested on independent data after the challenge deadline
- **Best participants will receive a winner's certificate and a prize!**
- **May the best machine learners win!**