The final project can be worked on in **groups of up to 3 students**. The purpose of this project is for you to you get hands-on experience on most topics of the course and to show that you can present and explain the results of your work. To get access to the template please use the following github link:
`https://classroom.github.com/a/HATfd6yP`.
You will have to write a paper of **max. 4 pages** with an optional appendix of **1 page per student in the team**. In the repository you can find a template for the paper. In the appendix the specific contributions of every student need to be specified. The paper needs to be submitted before August 31st 2022 to Openreview:
`https://openreview.net/group?id=automl.cc/AutoML/2022/Student/Conference`.
After submission every student will be assigned a paper to write a review before September 09th 2022 . Guidelines on how to write a review can be found in the repository. On September 15th & 16th 2022 we will organize a virtual conference on `https://app.gather.town/`. For the conference your group should create a poster and will have to present their results. The presentation should be 5 minutes and is followed by a 15 minute QA. After the conference you have until September 23rd 2022 to upload an improved and final version of your paper. Your grade is determined by the paper, poster, review, presentation and quality of code.

Joint Architecture Search and Hyperparameter Optimization of a Convolutional Neural Network

Your task is to automatically improve and analyze the performance of a neural network for a *fashion classification*[1] dataset. Instead of only considering the architecture and hyperparameters seperately you should build a system to **jointly optimize them**.
You are allowed a **maximum runtime of 6 hours**. We have provided a standard vision model as a baseline. In the end, you should convince us that you indeed improved the performance of the network when compared to the default approach. To this end, you could consider one or several of the following:

- (must) Apply HPO to obtain a well-performing hyperparameter configuration (e.g., BO or EAs);
- (must) Apply NAS (e.g., BOHB or DARTS) to improve the architecture of the network;
- (can) Extend the configuration space to cover preprocessing, data augmentation and regularization;
- (can) Apply one or several of the speedup techniques for HPO/NAS;
- (can) Apply meta-learning, such as algorithm selection or warmstarting, to improve the performance;
- (can) Apply a learning to learn approach to learn how to optimize the network;
- (can) Determine the importance of the algorithm's hyperparameters;

From the optional approaches (denoted by *can*), pick the ones that you think are most appropriate. To evaluate your approach please choose the way you evaluate well; you could consider the following:

- Measure and compare against the default performance of the given network;
- Plot a confusion matrix;
- Plot the performance of your AutoML approach over time;
- Apply a statistical test.

[1] `https://github.com/zalandoresearch/fashion-mnist`

You are allowed to use all scripts and tools you already know from the exercises; however, you are not limited to them. Overall, you should respect the following constraints:

- **Metric:**
  - The final performance has to be measured in terms of missclassification error.
- **Experimental Constraints:**
  - Your code for making design decisions should run no longer than 6 hours (without additional validation) on a single machine.
  - You can use any kind of hardware that is available to you. For example, you could also consider using Google Colab (which repeatedly offers a VM with a GPU for at most 12h for free) or Amazon SageMaker (which offers quite some resources for free if you are a first-time customer). *Don't forget to state in your paper what kind of hardware you used!*

As a starting point, we provide a repository containing:

- ToDo

General constraints for code submissions Please adhere to these rules to make our and your life easier! We will deduct points if your solution does not fulfill the following:

- We will use exclusively Python 3.7+.
- We expect Python scripts that conduct the experiments and creates results and visualizations.
- Add comments and docstrings, so we can understand your solution.
- (If applicable) The `README` describes how to install requirements or provides addition information.
- (If applicable) Add required additional packages to `requirements.txt`. Explain in your `README` what this package does, why you use that package and provide a link to it's documentation or GitHub page.
- (If applicable) All prepared unittests have to pass.
- (If applicable) You can (and sometimes have to) reuse code from previous exercises.

Grading:

- Paper: (at most 75):
  - Convincing motivation of the main idea in the introduction: 10 points
  - Sound and complete explanation of the approach: 20 points
  - Solution idea in general: 10 points for approaches from the lecture, further 10 points for ideas beyond the lecture
  - Thorough, insightful, and reproducible experiments: 20 points
  - Language quality (typos, gramma): 5 points
- Code (at most 25):
  - Well documented: 5 points
  - DocString: 5 points
  - Code quality: 5 points
  - Requirements: 5 points
  - Reproducibility[2]: 5 points
- Review (at most 25)
  - Reasonable summary of the paper at hand: 5 points
  - List of strong and weak points of the paper at hand: 10 points
  - Constructive feedback: 10 points
- Poster Presentation: (at most 60)
  - Well structured poster: 10 points
  - Clear message and insights: 15 points
  - Good illustrations, figures and plots: 15 points
  - Comprehensive short pitch (5min): 20 points

[2] `https://www.cs.mcgill.ca/~jpineau/ReproducibilityChecklist.pdf`