# Deep Learning for NLP 2022
# Homework 6

**Due on July 22, 2022 at 23:59**

TECHNISCHE UNIVERSITÄT DARMSTADT

Version of June 29, 2022

Overall: 25 points. This homework involves training more complex neural networks which can take some time. **Plan accordingly for the extended training time!**

Hint: You can check out Google Colab for "free" GPU access.

**Submission Guidelines**

This homework is structured with a Jupyter Notebook available in `https://colab.research.google.com/drive/1itlPTTplecvWWjpWMw5xsfnyYjGgBSV9?usp=sharing`. You can use Colab or your own hardware to complete the code and train the model.

```
hw6_group###.zip
 ├─hw6.pdf
 ├─hw6_colab.ipynb / hw6_cuda.py
 └─predictions_modelX.txt
```

- `hw6.pdf` should answer the questions **and** report your results.
- `###` is your group number (e.g., the file name could be `hw6_group2.zip`).
- `predictions_modelX.txt` should be a document that contains the prediction of your model.
- Please make sure that your code prints **all** relevant results for **all** subtasks when the tutors run `p6_colab.ipynb`.

Make sure that the results you report are reproducible and that your code is runnable in colab. Thank you!

## 1 Question Answering with Transformers (25P)

In this task, you will fine-tune a pretrained transformer-based architecture on a QA dataset with PyTorch and the Hugging Face library. You will also contribute to the community by sharing your model on the HuggingFace's Model Hub and will deploy it on UKP-SQuARE platform. With this, you will be able to use your model on a web interface, compare it with others, and analyze its behavior. Note: feel free to add or modify any method if that would make your code better.

### 1.1 Update conda environment (0P)

As training transformer-based models is computationally quite expensive, we propose two different settings that you can use to complete this homework. Depending on your individual resources, you can either train the model on your NVidia GPU or in Colab. If you use Colab, we recommend to write your code while using a CPU instance and change it to GPU once your code is ready to train. You will need to install the libraries: transformers, datasets, and huggingface_hub.

**Nvidia-GPU**

To avoid conflicts, please create a new conda environment with python version 3.7 (the environment name is just a suggestion):

conda create –name dl4nlp_hw6 python=3.7

Activate it and then follow these instructions:

1. Install PyTorch 1.11.0:
   conda install pytorch torchvision torchaudio cudatoolkit=11.3 -c pytorch
2. Install Huggingface Datasets:
   pip install datasets transformers huggingface_hub
3. Install huggingface_hub:
   pip install huggingface_hub
4. Install the libraries listed in the requirements.txt file from the originally provided dl4nlp_env.zip:
   pip install -r requirements.txt

**Google Colab**

Colab has many preinstalled packages, including all from the initial homework conda environment, with slightly different package versions. Therefore, you only need to execute the following line at the start of your Colab notebook:

!pip install transformers datasets huggingface_hub

In our case, this installed datasets 2.3.2, huggingface-hub 0.8.1, and transformers 4.20.1.

## 1.2 Data and Model Preparation (1P)

You MUST register the pretrained model and dataset you want to use for this assignment here: HW06 Model and Dataset selection.

We have selected seven pretrained models that are trainable in less than one hour in colab to ease your training:

- microsoft/xtremedistil-l6-h256-uncased
- microsoft/xtremedistil-l12-h384-uncased
- microsoft/xtremedistil-l6-h384-uncased
- distilbert-base-uncased
- microsoft/MiniLM-L12-H384-uncased
- huawei-noah/TinyBERT_General_4L_312D
- huawei-noah/TinyBERT_General_6L_768D

We have also selected six well-known QA datasets that have already been preprocessed to have the same format:

- SQuAD
- NewsQA
- TriviaQA
- SearchQA
- HotpotQA
- NaturalQuestions

In this homework, we will not use the test set because they are privately held by the authors of the original datasets, and using them would add an unneeded extra layer of complexity to this assignment. Hence, we will train on the training set and evaluate on the dev set. You can find their training and dev sets here: `https://github.com/mrqa/MRQA-Shared-Task-2019`.

**Tasks:**

1. Set the torch seed to a specific number so your results are reproducible.
2. Download and load your QA dataset.

   Hint: `https://github.com/mrqa/MRQA-Shared-Task-2019/blob/master/predict_server.py`
3. Load the pretrained model with a QA head.
4. Load the Tokenizer for your pretrained model.

## 1.3 Data Preprocessing (6P)

Now that we have loaded the data, the tokenizer, and the model, it is time to prepare the data for the model.

**Tasks:**

1. Do some data exploration to understand the format of the dataset. **Print** the first three examples of the training set and show their questions, contexts, and answers.
2. Encode the training and dev sets with the tokenizer, using a max length of **512**, truncation only on the context, and padding to max length.
3. Check out the predefined get_labels function and modify it, if necessary, to fit your data and return the correct start and end positions of the tokenized answer. Then complete the preprocess_function and apply it to the train and dev set.

## 1.4 Model Training (7P)

With the data prepared and ready to load, we can start to train our model.

**Tasks:**

1. Create the training loop and train your model. Use the following hyperparameters: learning rate: 1e-5, train batch size: 16, evaluation batch size: 32, epochs: 1, weight_decay: 0.01, optimizer: AdamW, number of warm up steps: 0.
2. Modify the hyperparameters to try a total of four additional different setups that outperform the provided default settings. You can modify the number of epochs, the batch size, and the learning rate. At least one of the settings has to **noticeably** outperform the default settings.

   Hint: adding epochs is the easiest way to increase the performance of your model.
3. **Plot** the loss value against training steps and display the plot for your best performing model in hw6.pdf.

## 1.5 Model Evaluation (7P)

Every time you train a model, you also need to evaluate its performance on an evaluation set. In this homework, we will use the dev set for this.

**Tasks:**

1. The raw outputs of the model are logits, not strings. Create a method that process the outputs of the model to create the answers to the questions in string format.
2. For each model you train, you need to evaluate it. In addition, submit the model predictions for all your trained models in a separate file. You should name the file as *predictions_model-hyperparams*, where hyperparams should be the hyperparameters you used to train the model AND that differ from the default settings. For example: predictions_distillbert (default), predictions_distillbert-2epochs, ... In your report, list your utilized hyper parameter settings and briefly justify (up to two sentences per setting) why you changed them from the default setting. Additionally, **print and report** your exact match (EM) and F1 score on the dev set after each epoch for each model.
3. If you are using more than one epoch, you should evaluate it after each epoch and plot a graph of the evolution of the EM and F1. Display the plot of your best performing model in hw6.pdf.

## 1.6 Sharing your Model (1P)

Now that you have trained and evaluated your models, it is time to make them available to the public. **Tasks:**

1. Create an account on HuggingFace. You can use your name if you want, or if you prefer more privacy, you can use a username following this pattern: *DL4NLP-GroupX*, where $X$ corresponds to your group ID.
2. Upload your best model. The name should follow this pattern: *pretrained_weights-dataset*. For example: xtremedistil-l6-h256-uncased-squad.
3. Write in the model card the hyperparameters you used, the EM and F1 scores of your model on the dev set AND your group ID. You don't need any code for this, just go to the model card in a web browser and edit the model card. Use this as template `https://huggingface.co/docs/hub/models-cards` and add as much metadata as possible since it will help your model be better positioned on the Model hub. **Print and report** the link to access your model.

## 1.7 Deploying your Model on UKP-SQuARE (3P)

As you published your best performing model, you can now use it for QA experiments on the `https://square.ukp-lab.de` platform.

**Tasks:**

1. Deploy your **best model**.
2. Run your model in the web interface of SQuARE and compare it with other extractive QA models. Take a screenshot of your model answering a question. This screenshot should contain your model and two other available models. You can write your own question and context or use any of the predefined examples of the available models. Display the screenshot in the hw6.pdf.

   It is okay if your model does not correctly answer all questions, the goal of this homework is not to make a state-of-the-art model, but to learn how to fine-tuned a QA model.