Task: GPT – Hand in, until 31.08.

- If something is underspecified, just make decision yourself
- Well-documented code
- Submission format
    - Notebook (incl. pdf) or GitHub readme (submit pdf with link to repo) as technical report of what we did
        - Nice narrative and way to navigate code, not scientific paper
        - Include plots (loss, perplexity scores, hyperparameters, etc.)
        - Optional include pseudocode
        - Qualitative analysis nice to have, e.g, add and evaluate generated text in report
        - Can add appendix for additional plots
- Hand in every mile stone, starting from **UNIX** comments
- Removed in-between milestone of causal-self attention
- Everything together in **one file**
- Compare the models from each milestone, report perplexity for all
    - Old-school n-gram
    - Best neural n-gram
    - GPT

GPT itself

- Hyperparameter tuning: do not need all of them, choose what is most interesting and explain why
    - Number of merges in BPE (not complete gridsearch, isolate top three number of merges in perplexity in n-gram, test those for GPT)
    - Regularisation
    - How small can we make neural embedding
    - Do not change optimiser
- General remarks
    - Transformer blocks from scratch would be **beyond 1.0**, not required
    - Implement causal self-attention yourself, do not use ready-made PyTorch version
    - For computing perplexity: Implementing teacher forcing annealing is necessary for good generation performance, but we don't **have** to do it for our assignment
- Reminders
    - Skip weight initialisation and optimiser configuration
        - Can use standard PyTorch initialisation → just get transformer parameters and add them when initialising the optimiser
    - Remember to change device selection, currently "cuda", you might want "mps" or "cpu"
    - Configs: make n_embd smaller, don't change betas and weight decay (unless you want to), can change batch size, chunk size, n_head, n_layer
    - Specify temperature and top-k parameters for generate function
    - Activation function used in MLP: not ReLU as in slides but GELU (might not be in PyTorch yet)