CS689 Machine Learning - Fall 2025

Homework 6

Due: Friday December 12th, 6:00 PM

Getting Started:

- **Course:** COMPSCI 689 Machine Learning, Fall, 2025
- **Instructor:** Justin Domke
- **Assignment:** 6
- **Group work policy:** No group work is permitted for this assignment. Verbal discussion with other students is acceptable, subject to conditions in the [Syllabus].
- **AI usage policy:** Limited use is acceptable, subject to conditions in the [Syllabus].
- **Due date:** Friday December 12th, 6:00 PM
- **Late submission policy:** Subject to conditions in the [Syllabus].
- **Submission instructions:**
  - For this assignment, you can prepare your solutions in any manner you like. You may use Markdown, LaTeX, handwrite them, or a combination of these. Regardless of how they are prepared, you must produce a single .pdf file for your report, and a zip file containing your source files and your code that you upload to Gradescope.
  - At the top of your .pdf file please include the following information:
    * Your name
    * Your email
    * (Please do **not** include your student ID.)
    * List anyone you discussed the assignment with, including course staff.
  - Additionally, you **must submit a .zip file** in Gradescope. Your .zip file should contain three things:
    * `report_src/` - A directory containing all source files (e.g. images) for the report.
    * `code/` - A directory containing Python (or other language) code for all parts of the assignment.
    * `code/run_me.py` - A single Python file (or an executable in other language) that will generate all figures/numbers included in your report.
  - **Gradescope instructions:** submit your **PDF** to [Gradescope], click on **Assignment 6**, then click on **Submit PDF** to upload a single pdf file, submit your **ZIP** to Gradescope, click on **Assignment 6 - Code** to upload a single zip file. After the pdf has been uploaded, mark the correct pages for each question. Contact Ke if you do not have access to the Gradescope page.
  - For the purpose of late days, the later of your two submissions will be considered the submission time for your assignment. E.g., if you submit your .pdf on time, but the .zip is two days late, the assignment will be considered two days late.
- **Gradescope policy**: Please ensure that all questions are properly marked on Gradescope. Failure to mark the correct pages will result in a warning for the first offense and a deduction of points for subsequent instances.
- **Note:** If you have any simple clarifying questions, you may use [Piazza] to ask your question. (Please let Ke know if for any reason you do not have access to the course Piazza forum.) For any significant questions, please come to the [office hours]. Because Piazza is meant for simple clarifying questions that can be answered in a paragraph at the most.

Mini-Project: Building a Small Language Model

Overview

In this mini-project, you will design, train, and evaluate a sequence of increasingly capable small language models (S-LLMs). Your goal is to investigate, empirically, how architectural components and computational budget affect model quality. You will begin with simple linear predictors and progressively introduce nonlinearities, self-attention, and transformer layers.

This project is intentionally open-ended, but your report must follow the required structure and include the specified plots and results. At the same time, you are encouraged to be **creative**: feel free to explore additional architectural ideas, training strategies, ablations, or comparisons beyond the ones listed here. Novel attempts—even those that do not improve performance—are still excellent for the purposes of this assignment.

Dataset

We will use several small, commonly used datasets for next-token prediction:

- **Tiny Shakespeare** A $\sim$ 1MB character-level dataset of Shakespeare's works.
- **WikiText-2** A word-level dataset derived from cleaned Wikipedia articles.
- **Penn Treebank (PTB)** A classic word-level dataset consisting of Wall Street Journal text.

Unless otherwise stated, you will use **Tiny Shakespeare with character-level tokenization** to train your initial models.

Training

You will build and train the following sequence of models on the **Tiny Shakespeare** dataset to predict the next character:

- **Linear predictor** A single linear (softmax regression) layer.
- **Multi-layer Perceptron (MLP)** At least 3 layers, with nonlinear activations.
- **Multi-head self-attention model** Implement one or more self-attention layers with configurable numbers of heads.
- **Multi-layer transformer** A small transformer with multiple blocks.

For each model family, you should experiment with:

- *Context length* (sequence length).
- *MLP hyperparameters*: number of hidden units, number of layers, activation functions.
- *Self-attention hyperparameters*: number of heads, head dimension.
- *Transformer hyperparameters*: number of layers, embedding size, MLP width.
- *Optimization choices*: learning rate, scheduler, optimizer (e.g. Adam vs. SGD), batch size.

You may also/instead **explore other design dimensions** if you wish, such as a combination of embedding, attention and MLPs for the architecture, alternative embedding strategies, positional encodings, regularization techniques, weight tying, or modified training objectives. These are entirely optional but encouraged if you are curious. If you do choose to experiment with other ideas instead, report at least three architecture design choices (instead of the model families in the template).

Deliverable 1: Tiny Shakespeare Experiments

For each architecture listed above, report:

0.1 The best hyperparameter settings you found.

0.2 A plot of **training loss vs. epochs**.

0.3 A plot of **test-set log-likelihood vs. at least three hyperparameter settings**. Examples: context length for the linear model, hidden dimension for the MLP, number of heads for self-attention.

0.4 A plot of **test-set log-likelihood vs. training FLOPs**.

0.5 A 100-character generation using prompt: `HAMLET:`

You are welcome to include **additional plots or exploratory comparisons**, even if they show that some ideas did not work as expected. In contrast to a research paper, negative results will still be considered valuable and can strengthen your analysis.

Deliverable 2: Word-Level Modeling (PTB and WikiText-2)

Using your best architecture from Deliverable 1, train on the **PTB** and **WikiText-2** word-level datasets. For each dataset, include:

0.1 A plot of **training loss vs. epochs**.

0.2 A plot of **test-set log-likelihood vs. training FLOPs**.

0.3 A 100-word sample generated from the model trained on **PTB**, using prompt:
`the school announced that`

0.4 A 100-word sample generated from the model trained on **WikiText-2**, using prompt:
`The history of machine learning begins`

You may **optionally** include additional prompts or qualitative evaluations if you find them interesting.

Report Requirements

The final report should include:

- **Introduction**: What architectures and settings you explored, and why.
- **Experimental Setup**: Datasets, tokenization choices, model architectures, and training procedures.
- **Results**: Required plots, tables, and any additional diagnostics.
- **Discussion**: Trends you observed regarding compute vs. performance, effect of context length, scaling behavior, etc. If you tried creative ideas (successful or not), describe what happened and what you learned.
- **Summary**: What you learned about designing efficient small language models.

Rubric (100 points)

- **Clarity and Organization (10 pts)**
  The report is clearly written, logically structured, and follows the required format. Explanations of modeling and experimental choices are coherent and easy to follow.
- **Experimental Correctness (30 pts)**
  Models are implemented properly, experiments are reproducible, and hyperparameters are documented. If you follow the template, the reported final test-set log-likelihood should fall within a reasonable range.
- **Plots and Analysis (30 pts)**
  All required plots are included, well-labeled. and easy to interpret. The accompanying analysis discusses observed trends and demonstrates a solid understanding of the results.

- **Depth of Exploration (15 pts)**
  The project includes a meaningful exploration of model variants or design choices. Creative or unusual ideas (even if unsuccessful) are valued here.
- **Insightful Discussion (10 pts)**
  Discussion includes thoughtful interpretation and interesting observations. Reflect on what you tried, what surprised you, and what failed.
- **Text Generation Quality (5 pts)**
  Generated samples meet a reasonable quality threshold.

Submission

Submit:

- A PDF report (3–6 pages).
- A ZIP file containing all your code (any language / framework).