CS 839: FOUNDATION MODELS
HOMEWORK 2

Instructions: Read the two problems below. Type up your results and include your plots in LaTeX. Submit your answers in two weeks (i.e., Oct. 21 2025, end of day). You will need a machine for this assignment, but a laptop (even without GPU) should still work. You may also need an OpenAI account to use ChatGPT, but a free account should work.

1. (30 pts) Prompting. We will attempt to see how ChatGPT can cope with challenging questions.

- 1. *Zero-shot vs. Few-shot.* Find an example of a prompt that ChatGPT cannot answer in a zero-shot manner, but can with a few-shot approach.
- 2. *Ensembling and Majority Vote.* Use a zero-shot question and vary the temperature parameter to obtain multiple samples. How many samples are required before majority vote recovers the correct answer?
- 3. *Rot13.* In this problem our goal is to use Rot13 encoding and 'teach' ChatGPT how to apply it. You can use rot13.com to quickly encode and decode. Also read about it at https://en.wikipedia.org/wiki/ROT13. Our goal is to ask questions like

  What is the capital of France?,

  but encoded with Rot13, i.e.,

  Jung vf gur pncvgny bs Senapr?,

  - What do you obtain if you ask a question like this zero-shot? Note: you may need to decode back.
  - What do you obtain with a few-shot variant?
  - Provide the model with additional instructions. What can you obtain?
  - Find a strategy to ultimately produce the correct answer to an encoded geographic (or other) question like this one.

2. (50 pts) NanoGPT Experiments. We will experiment with a few aspects of GPT training. While this normally requires significant resources, we will use a mini-implementation that can be made to run (for the character level) on any laptop. If you have a GPU on your machine (or access to one), even better, but no resources are strictly required.

- 1. *Clone Karpathy's nanoGPT repo* (https://github.com/karpathy/nanoGPT). We will use this repo for all the experiments in this problem. Read and get acquainted with the README.
- 2. *Setup and Reproduction.* Run the Shakespeare character-level GPT model. Start by running the prep code, then a basic run with the default settings. Note that you will use a different command line if you have a GPU versus a non-GPU. After completing training, produce samples. In your answer, include the first two lines you've generated.
- 3. *Hyperparameter Experimentation.* Modify the number of layers and heads, but do not take more than 10 minutes per run. What is the lowest loss you can obtain? What settings produce it on your machine?
- 4. *Evaluation Metrics.* Implement a *specific* and a *general* evaluation metric. You can pick any that you would like, but with the following goals: Your specific metric is meant to capture how *close* your generated data distribution is to the training distribution. Your general metric need not necessarily do this and should be applicable without comparing against the training dataset. Explain your choices and report your metrics on the settings above.

- 5. *Dataset.* Obtain your favorite text dataset. This might be collected data by a writer (but not Shakespeare!), text in a different language, or whatever you would prefer. Scrape and format this data. Train nanoGPT on your new data. Vary the amount of characters of your dataset. Draw a plot on number of training characters versus your metrics from the previous part. How much data do you need to produce a reasonable score according to your metrics?
- 6. *Fine-tuning.* Fine-tune the trained Shakespeare model on the dataset you built above. How much data and training do you need to go from Shakespearean output to something that resembles your dataset?

3. (20 pts) Beyond Transformers: Standalone and Hybrid Sequence Models. Recent research explores new architectures that move beyond standard Transformers, motivated by efficiency, longer context windows, or improved inductive biases. These include both *standalone* alternatives to attention and *hybrid* architectures that combine multiple paradigms.

- **Standalone models:**
  - **Mamba** [Gu and Dao, 2024]: a selective state-space model (SSM) achieving linear-time inference (*COLM 2024*).
  - **xLSTM** [Beck et al., 2024, 2025]: a modernized recurrent model with exponential gating and long-memory variants (*NeurIPS 2024*, *ICML 2025*).
  - **Hyena** [Poli et al., 2023]: a convolutional hierarchy enabling long-range implicit interactions (*ICML 2023*).
  - **Linear Attention** [Katharopoulos et al., 2020]: kernelized attention with subquadratic complexity (*ICML 2020*).
  - **Log-Linear Attention** [Guo et al., 2025]: a recent attention variant with logarithmic memory growth (*arXiv 2025*).
- **Hybrid models:**
  - **MambaFormer** [Park et al., 2024]: alternates Mamba and attention blocks for improved in-context learning (*ICML 2024*).
  - **Jamba** [Lieber et al., 2025]: mixture-of-experts hybrid combining Transformer and Mamba layers for long-context efficiency (*ICLR 2025*).
  - **Manticore** [Roberts et al., 2025]: a framework for composing pretrained Transformer and SSM components (*COLM 2025*).
  - **Samba** [Ren et al., 2025]: integrates Mamba SSMs with sliding-window attention for scalable long-context modeling (*ICLR 2025*).

You may also choose a different recent sequence model, such as **RetNet** [Sun et al., 2023], **RWKV** [Peng et al., 2023], **DeltaNet** [Yang et al., 2024], or **Griffin** [De et al., 2024]. If you choose another model, please provide a proper citation.

(a) *Conceptual (10 pts).* Pick one model and summarize its core mechanism and how it modifies or complements attention.

(b) *Tradeoffs (10 pts).* Discuss its motivation (e.g., efficiency, inductive bias) and potential drawbacks relative to Transformers.

(c) *Implementation (25 optional bonus pts).* Modify your NanoGPT implementation by replacing the self-attention module with a simplified version of your chosen model (e.g., a toy SSM or hybrid block). Train briefly on the same dataset and compare performance using your evaluation metrics from Problem 2. Comment on how the model's behavior differs across metrics or qualitative samples.

References

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael K. Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xlstm: Extended long short-term memory. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024. Spotlight.

Maximilian Beck, Korbinian Pöppel, Phillip Lippe, Richard Kurle, Patrick M. Blies, Günter Klambauer, Sebastian Böck, and Sepp Hochreiter. xlstm 7b: A recurrent llm for fast and efficient inference. In *Proceedings of the 42nd International Conference on Machine Learning (ICML 2025)*, 2025.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando de Freitas, and Caglar Gulcehre. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Proceedings of the 2024 Conference on Language Modeling (COLM 2024)*, 2024.

Han Guo, Songlin Yang, Tarushii Goel, Eric P. Xing, Tri Dao, and Yoon Kim. Log-linear attention. *arXiv preprint arXiv:2506.04761*, 2025.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*, volume 119 of *Proceedings of Machine Learning Research*, pages 5156–5165, 2020.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, Omri Abend, et al. Jamba: Hybrid transformer-mamba language models. In *Proceedings of the 13th International Conference on Learning Representations (ICLR 2025)*, 2025.

Jongho Park, Jaeseung Park, Zheyang Xiong, Nayoung Lee, Jaewoong Cho, Samet Oymak, Kangwook Lee, and Dimitris Papailiopoulos. Can mamba learn how to learn? a comparative study on in-context learning tasks. In *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*, 2024. Poster.

Bo Peng, Yizhe Mao, Yi Ding, Tri Dao, and Alexander M. Rush. Rwkv: Reinventing rnns for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. In *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*, volume 202 of *Proceedings of Machine Learning Research*, pages 28043–28078, 2023.

Liliang Ren, Yang Liu, Yadong Lu, Yelong Shen, Chen Liang, and Weizhu Chen. Samba: Simple hybrid state space models for efficient unlimited context. In *Proceedings of the 13th International Conference on Learning Representations (ICLR 2025)*, 2025. Poster.

Nicholas Roberts, Samuel Guo, Zhiqi Gao, et al. Pretrained hybrids with mad skills. In *Proceedings of the 2025 Conference on Language Modeling (COLM 2025)*, 2025.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024.