Course Project

Proposal and Final Submission Instructions

Project Proposals Due: July 31st by 11:59 pm
Final Projects Due: August 21st by 11:59 pm

For the remainder of the course your homework will be to complete a final project rather than weekly programming. There will still be occasional readings and short answers- look for these separately.

The final project will consist of implementing a variation or alternative to the GPT transformer architecture, tokenizer, training loop, or inference loop, and testing how this impacts the model. You may invent your own variation of interest or implement something from existing literature. You can also select one of the example ideas at the end of this document.

You should test your variation through metrics that show how your variation compares to the model we implemented together in previous homeworks. How do your changes impact training time, convergence, performance (or whatever the relevant factors may be)?

To assist in creating compelling project ideas, you will first submit a project proposal. You will be provided feedback on your individual project ideas and expectations for a successful final submission.

Note: You do <u>not</u> need to *improve* the existing GPT architecture (which has been optimized by the research community). You are simply tasked with implementing a specific variation of GPT models. It is okay if your implementation does not offer any change from (or even performs worse than) the GPT model we have already built, as long as your idea is well motivated and you can explain your results.

You may form groups of up to 3 students if you wish, but this is not required (can be individual). Groups can submit together, but will be held to a higher standard than individuals.

<u>NOTE:</u> The initial version of the syllabus had a draft of the project that was more along the lines of hyperparameter tuning / exploring relations between hyperparameters. This was revised because it requires many training runs to do properly, and training can take a long time. If you are interested in something like this *and* have access to several GPUs, let me know and we can discuss what this type of project may look like.

The final project will have the following components, which are each graded separately:

1. Project Proposal (20%): A written proposal including your intended variation of the GPT model, experiments to evaluate, and hypothesis as to the impact of the variation. Include all team member's names. The proposal must show critical thinking as to the variation implemented, why it may be significant, and how it will be evaluated.

The proposal is also an opportunity to get feedback on your ideas. You will be expected to incorporate proposal feedback into your final submission.

2. Programming Component (40%): A working version of your variation and code to use your implementation, with instructions to run. This will be graded similarly to other programming assignments (clarity, functionality, ease of use, etc). Your results should be reproducible by running your provided code.

3. Final Report (40%): A multi-page report detailing the relevant aspects of GPT models, your proposed change and its motivation, a comparison between your change and the version of GPT from homeworks, and conclusions that can be drawn from these results. This will follow a typical research paper / lab report structure as you may have seen in previous courses. See below for more details.

Proposal (Due July 31)

In a paragraph or two, please explain the following:

- Your project idea (What you intend to implement)
- How you expect your implemented component to differ from the GPT version from our homeworks (or if you are unsure, explain why this idea is of interest)
- Specific experiments or comparisons you intend to make. What are the key metrics that you are interested in?
- If applicable, any datasets or libraries you will use.

Include all team members' names (or state that the project is individual). As with the short answers, any typical text format is acceptable. Please upload to Canvas.

As an example, if your project was to implement binary BPE (instead of character-level BPE from homeworks), you may want to touch on things like:

- Expected utility of this change- this would allow support for many other languages and symbols
- Dataset- perhaps you try training on a corpus of text that is in a non-alphabetic language
- Comparisons- how would this compare to simply using <unk> for unsupported characters? How does the new tokenization impact things like vocabulary, training time, performance?

Project Programming (Due August 21)

An implementation of your project in code. In addition to your key files, this should include the following:

README.md (or README.txt) - a README file that explains your code and how it is intended to be used. Explain any outputs of your code (plots, terminal output, etc), and any additional scripts beyond main.py (for example, do you have a second script that performs analysis after training is complete?).

main.py - a file to run your key experiment(s), such as a training loop or tokenization routine.

installation.txt - Optional. Include instructions on how to install any libraries or access necessary datasets. You can alternatively include this in the README if you would prefer.

As with homework programming assignments, this should be well commented, formatted, and readable. It will be graded similarly to previous programming assignments.

An implementation of GPT as per previous assignments will be provided (including trained weights). You can use this or your own previous homework submissions as a starting point.

Project Report (Due August 21)

Please compile a paper / lab report on your project that explains the motivation behind your project, what you implemented, and how you found it to change the behavior or performance of a GPT model. You can use (and should include!) diagrams, tables, or plots as necessary. Your report should be roughly 3-5 pages single-spaced including figures, tables, etc (this is not strict but please don't write a novel). Specific formatting beyond that is up to you.

Please include the following sections in your document:

1) Introduction: Summarize your project. What did you implement and why? What was the expected outcome of your changes to the GPT model, or what functionality were you trying to add? What were your key results and conclusions?

2) Background: Explain the relevant mechanisms of the GPT model in your own words, and how your implementation is intended to differ. If someone knew about LLMs at a cursory level (and had general ML knowledge), what would they need to understand about your project? What are some relevant works or sources you consulted (if applicable.) You do not need to have a formal "related works" section as in a full academic paper. Just give the reader an idea of what we are talking about here.

3) Methods: Explain in detail how your project works, any datasets or libraries used, and specific experiments you performed or metrics you used. In short- what did you make and how does it work?

4) Results: After implementing everything above and running it, what did you find? Include any numbers, tables, plots, etc. If you had qualitative comparisons (i.e. inspected the generation of two models side by side), include some examples.

5) Conclusions: What did you learn from this project? What can we conclude from your results? You can also take this opportunity to mention any obstacles that you faced or surprises along the way- did something work exceptionally well (or not)?

6) Impact and Future Work: How might your results impact large-scale models that may be trained on 1000s of GPUs and/or released to the masses? If you had access to those types of resources, what would you want to try next?

You do not need to use LaTeX, but please submit a nicely formatted document (word, google doc, pdf, etc) that includes figures in the body. A plain text file is not sufficient for this submission.

Note on Team Submissions

If you work in a team, you only need to submit once with all team members' names clearly indicated. Any team member can submit. Teams will be held to a higher standard than individuals. If you are interested in implementing something very ambitious, I would encourage working in a team (and I will discourage individual projects that seem too complex to implement in a short timeframe).

Example Project Ideas

You may choose something from this list to implement if you wish, but you are also free to come up with your own ideas in a similar vein. These are organized by topic:

- Tokenizers:
    - Byte-level BPE
    - WordPiece
    - SentencePiece
- Embeddings:
    - Rotary Position Embeddings
    - Word2Vec
- Attention:
    - Sliding window attention / longformer
    - One of the many other transformer variants
    - Cross attention
- Transformer Model:
    - Exploration of model architecture settings/sizes*
    - Your own variant of the transformer block (would need to be well motivated)
- Training:
    - Exploration of training hyperparameters*
    - Exploring / defining a new scaling law for some aspect of the GPT model*
- Inference:
    - Inference with KV-Caching
    - Custom sampling strategy
- Later Lectures or Other:
    - Training a code generation model
    - Fine-tuning for a specific downstream task
    - Parallelized training or inference*

*These would require more compute than the others.