Decoding GPT: Intro Project

Assignment

1. train a markov chain on a large text corpus, but using the gpt-2 tokenizer
   - see code from existing notebooks for how to get and use this tokenizer from the huggingface `transformers` package
2. investigate the sparsity of the resulting matrix in a more rigorous way than just the eyeball norm:
   - plot histograms of the weights
   - try to find the rank
   - take the SVD, and plot the singular values
   - other matrix decompositions or measures?
3. do some kind of investigation of your choice, and produce a legible juptyer notebook with your results. some possible questions might be:
   - can you *train* the sparse representation directly? (maybe using pytorch's autograd?)
   - are there any interesting clusters in the matrix? (hint: PCA, k-means, t-SNE, etc)
   - can you come up with sparse representations of trigrams? n-grams?
   - how do the results change if you use a different tokenizer? (i.e. our original word-based one)

general guidelines

- feel free to discuss in groups, but everyone should submit their own notebook
  - exceptions can be made for ambitious projects, send Michael an email first though
- use any code from notebooks or the internet, but cite your sources
- use ChatGPT or other LLM tools, but:
  - you should understand and be able to explain what is happening in the produced code
  - document the prompts you used!

- it's ok if what you try doesn't work!
  - show me your code, make it understandable, and explain what you learned!
  - what would you do differently in retrospect?

This project is intended to be very open ended, and get you comfortable with writing code to investigate an artifact that is the result of some kind of training process. If you decide work on something significantly different from the stated problem domain, this is fine! (Just check with Michael if you haven't done so already).

Submission format

The default submission should be a single, self-contained jupyter notebook with both your explanation of the techniques you are trying, your code, and discussion of the results. For larger projects with more than one file, submit either a zip file with all contents or a link to a github repo.

- if you use packages outside the ones in our default `requirements.txt` file, include your own `requirements.txt` file. It should be possible to install the requirements from this file and run the notebook without issues.
- your code should be legible, and contain sufficient comments to explain what is happening
  - I encourage wrapping operations into functions whenever possible!
- if your notebook has a lot of code, feel free to export things to external `.py` modules – just make sure you include them in your submission!
  - If your submission contains more than just a single notebook, please include a `readme.md` file detailing the general structure of your submitted files and telling the reader what to look at
- I *strongly encourage* using version control for your project, so you can go back to previous versions if you break things!