CSCI 1051 Problem Set 2

January 14, 2025

Submission Instructions

Please upload your solutions by **5pm Friday January 17, 2025.**

- You are encouraged to discuss ideas and work with your classmates. However, you **must acknowledge** your collaborators at the top of each solution on which you collaborated with others and you **must write** your solutions independently.
- Your solutions to theory questions must be written legibly, or typeset in LaTeX or markdown. If you would like to use LaTeX, you can import the source of this document here to Overleaf.
- I recommend that you write your solutions to coding questions in a Jupyter notebook using Google Colab.
- You should submit your solutions as a **single PDF** via the assignment on Gradescope.

Problem 1: Visualizing Word Embeddings

In this problem, we will embed words using a pretrained embedding model. I recommend using the Pytorch `AutoTokenizer` and `AutoModel` with the "bert-base-uncased" pretrained weights.

Part A: 10D Visualizations

Embed (at least 8) words of your choice then create a visualization of the first ten embedding dimensions for each word.

Part B: 2D Plots

Plot the words in the first two embedding dimensions on a 2D plot. Do you notice any patterns?

Part C: 2D Plots via PCA

Use PCA to the embeddings and plot the resulting two principal components on a 2D plot. Do you notice any patterns?

Part D: Word Math

Try adding and subtracting word embeddings from each other, then plot the results in the 2D plots from Part B and Part C, respectively. Is the word you create close to what you imagined it would be?

For example, we may expect that

$$\text{king} - \text{man} + \text{woman} \approx \text{queen}.$$

Problem 2: Visualizing Attention

Part A: Self-attention Activations

Consider a pretrained transformer model. Select a text input of your choice. Plot the self-attention weights at several layers (both early and later) in your transformer model. What patterns do you notice?

Part B: Cross-attention Activations

Consider a pretrained transformer model for translation. Select a text input of your choice and translate it. Now feed the original and translated text to a translation model and plot the cross-attention weights at several layers (both early and later). What patterns do you notice?

Problem 3: Low Rank Adaptation

In this problem, we will investigate the advantages and disadvantages of low rank adaptation.

Part A: Architectures

Build a neural network with large linear layers (≥ 1 million parameters). In addition, build a similar neural network architecture where each layer $d_{\text{in}} \times d_{\text{out}}$ is replaced by two linear layers of dimensions $d_{\text{in}} \times r$ and $r \times d_{\text{out}}$ for a small rank $r \approx 100$.

Part B: Comparison

Train instances of both models on a dataset of your choice. Compare the number of parameters in each model and the time it take to train each model. Finally, plot the test loss by epoch for the two models.

Problem 4: Watermarking

In this problem, we will implement and compare two of the watermarking schemes we discussed in class.

Part A: No Watermarking

Write a method that takes an input string `text` and auto-regressively samples `num_tokens` from an LLM of your choice.

Part B: Red/Green Watermarking

Write a method that samples from the red/green watermarking scheme. Write a corresponding method that detects whether a text input was likely watermarked with the scheme.

Reminder: Fix the list of red/green words *once*.

Part C: Exponential Minimum Sampling

Write a method that samples from the exponential minimum sampling scheme. Write a corresponding method that detects whether a text input was likely watermarked with the scheme.

Reminder: Hash a few of the prior tokens so that you can reproduce the same uniform vector.

Part D: Comparison

Using the same text prompt, generate an output from each of the three generation methods you implemented above. For each output, check the detection costs for both of the detection methods you implemented above.

Repeat the above process with two text prompts; one should be a common phrase and the other should be rare. What do you notice?