L173 Exercise 2
Learning to unmask in a text diffusion model

D.J. Wischik

Your answer should be roughly 800 'prose words' (not including equations, tables, plots or code). As in exercise 1, it should be mathematically self-contained: define every symbol you use. You may quote results from the model solution to exercise 1 without proof. LLM usage is permitted.

Consider the text diffusion model of exercise 1. At each generation step, we will choose a fixed number $\ell > 1$ of positions to unmask. As in exercise 1, a predictor network reads the current string and returns a distribution over tokens for each masked position; each position chosen for unmasking is filled by an independent draw from its distribution.

1. Consider using a trainable network to choose which positions to unmask. This network should take in the current string, and probabilistically choose ℓ positions to unmask. Describe a form for this network and explain how to draw a random set of positions from it. Your network should be 'wired up' so that, given sufficient capacity, it could represent any distribution over sets of size ℓ.

2. Write down the bound from exercise 1 part 5 for the combined predictor and unmasker. We wish to train both of them by gradient descent. Find the gradient of this bound with respect to θ (the parameters of the predictor) and with respect to ϕ (the parameters of the unmasker). Explain how to estimate these gradients using sampling. *[The handout for lecture 3 may be useful.]*

3. Train the predictor and unmasker together, on a small text dataset[1] of your choice. Also train the predictor with the random unmasker, which selects ℓ positions uniformly at random. Report a holdout comparison; say briefly what your comparison measures. *[You will be marked for how you explain your method and results, not for your code nor the size of your run nor the success of your generator. Submit your code alongside your answer.]*

4. Does your trained unmasker bear out the suggestion you made in exercise 1 part 6? Describe how you checked.

[1] such as TinyStories, Eldan and Li (2023), `https://arxiv.org/abs/2305.07759`.