Homework 4

CS 521: Fall 2025

Due on Friday, November 7, 2025 11:59 PM Central

Homework Policy: You are allowed to collaborate with your classmates, but report this in your submission. Even if you work on problems together, each student must write up their solution individually in their own words in their submission. This CS 473 course page on academic integrity is a handy reference: `https://courses.engr.illinois.edu/cs473/sp2023/integrity.html`. We have ZERO tolerance for breaches of academic integrity, such as, but not restricted to, plagiarism, use of LLMs except for when stated explicitly, etc.
You can take upto 3 days extra in total across the semester without explanation. Once you exhaust the three-day late submission allowance, any further late homework submissions will result in a 25% penalty per each extra day. 5 minutes late in submitting the homework will be counted as 1 day late. If you have exceptional personal circumstances that will prevent you from submitting the assignment on time, write to the course staff as soon as possible. Unless explicitly stated, use of LLMs for solving homework or cheating from others is not allowed – we will detect it! There will be an oral **exam** if we detect violations, during which you will be asked to answer a randomly selected question from the homework. If you are unable to show sufficient knowledge, then you will get zero for the full homework.
Please format your submission as a PDF (file naming convention: *CS521_⟨netid⟩_hw2.pdf*) and upload it on Gradescope by the deadline. Try to start every problem on a new page, to accurately map pages to questions on Gradescope.
Programming Problems: Read these guidelines carefully! Describe your solution at a high-level (if applicable). Write only small snippets of code, if at all. DO NOT paste tall walls of code into your submission! Upload your solution files to GitHub publicly and provide a link to the relevant files as part of your solution.

Points: Problem 1: 10 + 20 = 30 points, Problem 2: 10 + 15 + 15 = 40 points, Problem 3: 15 + 15 = 30 points. Total: 100 points

Problem 1 (Best-of-N Sampling)**.** To align language models with human intent, reward models are trained to predict the human preference. The reward model's predictions serve as a proxy for human preference. Best-of-N sampling means that we use a model to sample N times and take the sample that scores the highest according to the proxy objective. In this question, we will use best-of-N sampling to get the expected reward using different N values. Some starter code can be found at `https://github.com/enyijiang/CS521FA25HW/blob/main/hw4/bon.ipynb`. Here we use a very simple model with uniform distribution between 0 and 4.

(a) (10 points) We have the reward function $R(x) = 5 - \|2 - x\|$. Plot the function R.

(b) (20 points) Plot $\mathbb{E}[R(\pi^{BoN})]$ (the expectation over 100 times of running best-of-N algorithm) with $N = 1, 2, 4, 8, 16, 32, 64$ and report any observations you have.

Problem 2 (Best-of-N Continued with Reward Hacking)**.** In this question, we will use some proxy reward function with reward hacking based on problem 1 and see the impact of reward hacking when using Best-of-N sampling. Some starter code can be found at `https://github.com/enyijiang/CS521FA25HW/blob/main/hw4/bon.ipynb`.

(a) (10 points) For the proxy reward function $\hat{R}$, compared with the reward function R, we add $noise = 2$ when $x \in [0, 0.01]$ or $[3.99, 4]$. Plot $\hat{R}$.

(b) (15 points) Compute the KL divergence between the initial distribution and the optimized distribution by using Best-of-N sampling with $N = 1, 2, 4, 8, 16, 32, 64$, you can either compute it analytically or numerically.

(c) (15 points) Plot $\mathbb{E}[R(\pi^{BoN})]$ and $\mathbb{E}[\hat{R}(\pi^{BoN})]$ (the expectation over 100 times of running best-of-N algorithm) with
$N = 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048$ and report any observations you have (e.g. do you observe the reward hacking and when it becomes more severe). Why choosing a large N for the proxy reward model would be a bad idea?

Problem 3 (Explanations, programming)**.** In this problem, we ask you to implement some explanation methods for the predictions of PyTorch's pretrained ResNet-18 model `https://pytorch.org/vision/main/models/generated/torchvision.models.resnet18.html` on some samples of the ImageNet [DDS$^+$09] dataset. In particular, you should implement the i) LIME [RSG16] and ii) SmoothGrad [STK$^+$17] methods that we have discussed in the lectures. Using implementations of explanation methods from open-source libraries is strictly prohibited and will be awarded 0 points. You are asked to generate the explanations for the model's predictions for 5 inputs from ImageNet given here: `https://uofi.box.com/s/8ndvhwwcgu94phios6t9x9q5hp9ri7hx`. You should provide qualitative analysis of the explanations (compare with human understanding of important parts of the images) and use the explanations to justify the correct/incorrect model predictions. You should also compare the explanations from the two explanation methods both qualitatively and quantitatively by providing the Kendall-Tau and Spearman Rank correlations between the respective feature rankings in the explanations. Some starter code for inference from ResNet-18 is here: `https://github.com/enyijiang/CS521FA25HW/tree/main/hw4`.

Bonus Research Question (25 points). In this question, we ask you to leverage an open-source RL package (e.g., `https://github.com/huggingface/trl`) to explore the reward-hacking effect on LLMs.

(a) (20 points) Compare the model you got with/without reward hacking and report any observations or findings you got.

(b) (5 points) Briefly explain why the PET [XKS+25] method can help mitigate reward hacking problem.

References

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “why should i trust you?”: Explaining the predictions of any classifier, 2016.

[STK+17] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise, 2017.

[XKS+25] Yinglun Xu, Hangoo Kang, Tarun Suresh, Yuxuan Wan, and Gagandeep Singh. Learning a pessimistic reward model in rlhf. *arXiv preprint arXiv:2505.20556*, 2025.