CSE Ph.D. Qualifying Exam, Spring 2026: Data Analysis

Instructions:

- This is a CLOSED BOOK exam. No books or notes are allowed.
- Please answer three of the following four questions. All questions are graded on a scale of 10. If you answer all four, all answers will be graded and the three lowest scores will be used in computing your total.
- Please write clearly and concisely, explain your reasoning, and show all work. Points will be awarded for clarity as well as correctness.
- Good luck!

1. **Q1: Reinforcement Learning**
Alice is asked to conduct reinforcement learning from human feedback (RLHF) on large language models(LLMs), which is denoted as $\pi_{ref}(y|x)$ with x as the prompt and y as the responded text. Please help Alice to complete the task.

   (1) [4 points] The first step of RLHF is to learn a reward model. However, there is no directly reward labeled, but only pairwise comparison provided by human, *i.e.*, given two answers y_1 and y_2 to the same prompt x, a human labeler will distinguish which one is better and which is worse, denoted as y_+ and y_-, respectively. Alice exploits the Bradley-Terry (BT) model to capture the pairwise comparison with an implicit reward $r_\theta(x, y)$, where θ denote the parameters of the model, *i.e.*,

$$\mathbb{P}(y_+ \succ y_-|x, y_+, y_-) = \frac{\exp(r_\theta(x, y_+))}{\exp(r_\theta(x, y_+)) + \exp(r_\theta(x, y_-))} \tag{1}$$

   With such data collected $\mathcal{D} = \{(y_+, y_-, x)_{i=1}^n\}$, please help Alice to design the maximum log-likelihood estimation of the BT model to learn θ.

   (2) [4 points] With the learned reward model, Alice would like to update the LLM $\pi_\omega(y|x)$ through policy optimization, where ω is the parameter, *i.e.*,

$$\max_{\pi_\omega} \ \ell(\omega) = \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\omega(y|x)}\left[r_\theta(x, y)\right] - KL(\pi_\omega||\pi_{ref})\right]. \tag{2}$$

   Please help Alice to derive the policy gradient w.r.t. ω, *i.e.*, $\nabla_\omega \ell(\omega)$. (Hint: you may need to use log-derivative trick here.)

   (3) [2 points] Alice was told she needs to repeat the step 1 and 2 iteratively. Could you please explain the reason to Alice?

2. **Q2: Variational Autoencoders and the Evidence Lower Bound (ELBO).**
Consider a latent variable model with latent variable $z \in \mathbb{R}^L$ and observed data $x \in \mathbb{R}^D$:

$$p_\theta(x, z) = p(z)\, p_\theta(x \mid z), \qquad p(z) = \mathcal{N}(0, I).$$

Let $q_\phi(z \mid x)$ denote a variational posterior used to approximate the true posterior $p_\theta(z \mid x)$.

(a) ELBO Derivation [3 pts]

Starting from the marginal log-likelihood $\log p_\theta(x)$, derive the Evidence Lower Bound (ELBO)

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x \mid z)] - \mathrm{KL}\big(q_\phi(z \mid x) \,\|\, p(z)\big).$$

Clearly state the inequality used and identify the conditions under which the bound is tight.

(b) Connection to EM and Variational Inference [4 pts]

Explain how the ELBO relates to the EM algorithm. In particular:

- Identify the analog of the E-step and M-step in variational inference.
- Explain why EM corresponds to exact posterior updates in some models, while VAEs rely on approximate inference.

Hint: Recall that EM can be viewed as coordinate ascent on a lower bound of $\log p_\theta(x)$. Compare which variables are optimized in the E-step and M-step with those optimized in variational inference.

(c) Gaussian VAE: Closed-Form KL Term [3 pts]

Assume the variational posterior is Gaussian:

$$q_\phi(z \mid x) = \mathcal{N}\big(z \mid \mu_\phi(x), \mathrm{diag}(\sigma_\phi^2(x))\big).$$

Derive a closed-form expression for $\mathrm{KL}\big(q_\phi(z \mid x) \,\|\, p(z)\big)$. Explain why this term acts as a regularizer on the latent representation.

3. **Q3: Adversarial Machine Learning**
In safety-critical applications, machine learning models must be robust against malicious actors. This problem explores the fundamental min-max formulation of adversarial training, the construction of attacks using constrained gradients, and the design of defensive strategies.

(a) [3 pts] In standard empirical risk minimization, we minimize the average loss over a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$. In adversarial training, we assume an attacker can perturb each input $x_i \in \mathbb{R}^n$ by a vector $\delta_i \in \mathbb{R}^n$ within a constraint set Δ. Please formulate the training objective of the dataset $\mathcal{D}$ for a model with parameters θ as a minimax optimization problem.

(b) [3 pts] To solve the inner maximization of the minimax problem, we often use Projected Gradient Descent (PGD). Let the constraint set be an ℓ_∞-ball of radius ϵ, defined as $\Delta = \{\delta : \|\delta\|_\infty \leq \epsilon\}$. Please describe how can the adversary use projected gradient descent to construct an adversarial example for every training instance. Please provide the pseudocode and analyze the computation cost.

(c-1) [2 pts] **Defense Strategy 1: Adversarial Training.** Explain how you would modify a standard training loop to incorporate the attack derived in part (b). Why does training on adversarial samples help solve the minimax objective?

(c-2) [2 pts] **Defense Strategy 2: Gradient Obfuscation.** Some defenses make the loss function L non-differentiable or "shattered" such that $\nabla_x L$ is zero or unreliable. Explain why this might cause the PGD attack to fail. However,

this defense strategy is usually considered fragile. Please design an algorithm to bypass the gradient obfuscation assuming the attacker is aware of the obfuscation mechanism.

4. **Q4: Contrastive learning**

Contrastive learning is a technique for learning representations by comparing an *anchor/query* example against multiple *key* examples. For example, in CLIP-style training, an *image* is embedded as a vector q (the query/anchor), and a set of *captions* are embedded as vectors $\{k_j\}$ (the keys). For a given image, the caption that truly describes the image is the *positive key* k^+, while other unrelated captions in the batch are *negative keys* $\{k^-\}$. The goal is to learn an embedding space where q is *close* to its positive key and *far* from negative keys.

A standard loss used to achieve this is the InfoNCE loss. This problem helps you derive that view and understand why the loss tends to pull positives together and push negatives away.

Let an anchor be $q \in \mathbb{R}^d$. You are given one *positive* key $k^+ \in \mathbb{R}^d$ and $K-1$ *negative* keys $\{k_2^-, \ldots, k_K^-\} \subset \mathbb{R}^d$. Define keys $\{k_1, \ldots, k_K\}$ by

$$k_1 = k^+, \qquad k_j = k_j^- \ \text{ for } j = 2, \ldots, K.$$

Define the (temperature-scaled) similarity scores with a given temperature $\tau > 0$ as

$$s_j \triangleq \frac{1}{\tau} q^\top k_j, \qquad \tau > 0.$$

Define the softmax probabilities

$$p_j \triangleq \frac{\exp(s_j)}{\sum_{i=1}^K \exp(s_i)}.$$

The InfoNCE loss (for this single anchor q) is defined as

$$\mathcal{L}(q) \triangleq -\log \frac{\exp(s_1)}{\sum_{j=1}^K \exp(s_j)}.$$

(a) [1 pt] Show that $\mathcal{L}(q)$ can be written as the cross-entropy loss of a K-class softmax classifier where the correct class is the positive key k_1. (In your answer, clearly specify the logits and the predicted probabilities.)

(b) [3 pts] Compute $\frac{\partial \mathcal{L}}{\partial s_j}$ for $j = 1, \ldots, K$ in terms of $\{p_j\}$. *Hint:* First show that $\mathcal{L}$ can be writtent as $-s_1 + \log\left(\sum_{i=1}^K e^{s_i}\right)$.

(c) [3 pts] Derive $\nabla_q \mathcal{L}(q)$ and express it in a simple form involving k^+ and a weighted average of keys $\{k_j\}$. Then interpret the result: why does optimizing InfoNCE tend to pull q toward k^+ and push q away from negatives? *Hint:* Use the chain rule and your result in (b).

(d) [3 pts] Breifly answer the following:

(d1) [1 pt] State what happens to the distribution $\{p_j\}$ in each limit: (i) as $\tau \to 0$, and (ii) as $\tau \to \infty$. Then add *one sentence* explaining which keys dominate the gradient in each limit and why.

(d2) [2 pts] If the set of negative keys contains *hard negatives* (i.e., some k_j with large similarity $q^\top k_j$ to the query), what qualitative effect do they have on the gradient magnitude, and why?