Homework 4: Reasoning, Vision-Language Models, and Agents

CS 388: Natural Language Processing • Fall 2026

Instructions. Please transcribe your answers/work into the provided LaTeXfile and submit the PDF. **Please use the \B{} command to wrap your solutions, making them blue (this will help us when grading).**

1. Question 1: RLVR and GRPO (35 points)

Below is one training step of a simplified version of GRPO for teaching a model to solve math word problems with reasoning. For simplicity, it leaves out the ratio clipping and the KL penalty to a reference model; you do not need to add them. The verifier is supposed to give reward 1 if the *final answer* of a completion is correct and 0 otherwise.

There are **two** logical mistakes in this code: places where the algorithm deviates from GRPO as we saw it in class. The code runs without errors. The errors are logical errors, as opposed to coding errors: you can assume, for example, that the helper functions (`generate`, `extract_answer`, `log_prob`) do what their comments say, and that `mean` and `std` compute the mean and standard deviation of a list.

```
1  def grpo_step(policy, optimizer, batch, G=8):
2      # batch: list of (question, answer) pairs, e.g. ("5 pens cost $10. What do 3 cost?", "6")
3      optimizer.zero_grad()
4      total_loss = 0.0
5
6      for question, answer in batch:
7          # sample a group of G completions (reasoning followed by a final answer)
8          completions = [policy.generate(question, temperature=1.0) for _ in range(G)]
9          # extract_answer(c) returns the final answer at the end of completion c
10
11         # verifier
12         rewards = [float(answer in c) for c in completions]
13
14         # group-relative advantages
15         m, s = mean(rewards), std(rewards)
16         advs = [(r - m) / (s + 1e-8) for r in rewards]
17
18         for c, adv in zip(completions, advs):
19             # log_prob(y, given=x): sum of the log-probs of the tokens of y given x
20             logp = policy.log_prob(answer, given=question)
21             total_loss = total_loss - adv * logp
22
23     (total_loss / (len(batch) * G)).backward()
24     optimizer.step()
```

(a) **(5 pts)** Find the first mistake. Give the line number(s) and a fix (pseudocode is fine).

(b) **(7 pts)** Explain the effect of this mistake. What would happen if we failed to fix it (ignoring the other error)? Describe concretely what kind of completions the model would learn to produce.

(c) **(5 pts)** Find the second mistake. Give the line number(s) and a fix.

(d) **(7 pts)** Explain the effect of this mistake. What would happen if we failed to fix it (ignoring the other error)? *Hint:* look at what the loss for a single question adds up to.

(e) **(6 pts)** Now assume both mistakes are fixed. For a question where all $G = 8$ completions are wrong, what are the advantages, and how much does this question contribute to the gradient? What about a question where all 8 completions are correct?

(f) **(5 pts)** You start GRPO training from a base model that solves essentially 0% of the training questions (none of its sampled completions are ever correct). Based on (e), what happens during training? Suggest one way to fix this.

2. Question 2: Vision-Language Models (30 points)

(a) **(12 pts) Designing a VLM application.** A city runs a 311 app where residents can submit a photo of a road problem. The city wants to automatically (1) decide whether the photo actually shows a pothole, (2) rate its severity as *minor*, *moderate*, or *severe*, and (3) write a one-sentence description for the repair crew (e.g. *"Large pothole in the right lane, next to a storm drain"*). Describe how you would address this with a VLM. Your answer should cover (3 pts each):

(i) what the model's input and output look like, including the prompt and the output format;

(ii) whether you would use the VLM zero-shot (prompting only) or fine-tune it, and what data you would need;

(iii) how you would evaluate the system before deploying it;

(iv) one realistic failure mode, and how you would mitigate it.

(v) **Ungraded coding exercise:** Implement this using a public dataset like `https://www.kaggle.com/datasets/chitholian/annotated-potholes-dataset` – you may need to add some annotations yourself.

(b) **(18 pts) Reviewing an architecture.** You are running an engineering team building a VLM for pharmacies. A pharmacist places pills on a counting tray and takes a photo from above with a fixed camera; the model must answer *"How many pills are on the tray?"* The photos are 2048×2048 pixels, each pill is roughly 40×40 pixels in the photo, and a tray holds up to 200 pills, often touching each other.

Engineer B proposes the following model:

Component	Setting
Image preprocessing	Resize each photo to 448×448 pixels
Vision encoder	ViT with patch size 64×64; 24 layers, hidden size 1024, 16 attention heads; pretrained with a CLIP-style objective on 400M image–caption pairs
Projector	2-layer MLP mapping each patch embedding to one LLM input token
Language model	7B-parameter decoder-only LLM (32 layers, hidden size 4096), context length 4096
Fine-tuning data	50k tray photos with the correct count, formatted as question–answer pairs
Optimization	AdamW, learning rate 2×10^{-5}, batch size 256, 3 epochs

(i) **(6 pts)** How many visual tokens does the LLM receive for each photo? How many pixels of the *original* photo does one patch cover (along each side), and roughly how many pills could fall inside a single patch? Show your work.

(ii) **(6 pts)** Do you let Engineer B train this model? Why or why not?

(iii) **(6 pts)** If not, what would you change? Be specific (give numbers).

3. Question 3: Agents (35 points)

A user asks an LLM agent:

"Schedule a one-hour meeting next week with Alice, Bob, and Carol."

The agent can read the user's calendar and the calendars of Alice and Bob, which are shared with the user. Carol's calendar is private. The agent can also look up Carol's *free/busy status* for a given time, send emails to any of the attendees and read their replies, and send a calendar invitation. The episode ends when the agent sends an invitation or gives up. The agent receives a reward of $+1$ if it sends an invitation for a time when all four people can attend, -1 if it sends an invitation for a time when someone cannot attend (the meeting will have to be rescheduled), and 0 if it sends no invitation.

(a) **(10 pts)** Write this task as a POMDP. Describe each component (states S, actions A, observations O, transitions $T(s' \mid s, a)$, and reward R) in a sentence or two, specific to this task.

(b) **(6 pts)** Name two parts of the state that the agent cannot observe directly. For each, name an action that gives the agent information about it, and say whether that information is complete or only partial.

(c) **(9 pts)** The agent finds that Tuesday at 2 pm is free for the user, Alice, and Bob. Based on what it knows, it estimates that Carol is free at that time with probability $p = 0.7$. The agent can either:

- **Invite now.** It sends the invitation for Tuesday at 2 pm.
- **Email Carol first.** Emailing costs a reward of c. Assume Carol answers truthfully. While the agent waits for her reply, there is a 0.2 probability that the slot is taken on someone else's calendar. If Carol is free and the slot is still open, the agent sends the invitation; otherwise it gives up.

(i) With $c = 0.1$, compute the expected reward of each option. Which should the agent choose?

(ii) What is the largest cost c for which emailing Carol first is still the better choice? (This is the *value of information* of Carol's reply.)

(d) **(10 pts)** Instead of emailing, the agent first looks up Carol's free/busy status for Tuesday at 2 pm, which is free of cost. The lookup is imperfect: it only knows about events that Carol has put in her calendar. If Carol is actually free, the lookup always shows *free*. If she is actually busy, it shows *busy* with probability 0.6 and *free* with probability 0.4 (she has not entered the event).

(i) The lookup shows *free*. Starting from the prior $p = 0.7$, compute the agent's updated belief that Carol is free.

(ii) With this updated belief (and still $c = 0.1$), should the agent invite now or email Carol first? Show the expected rewards.

(iii) What should the agent believe, and do, if the lookup shows *busy* instead?