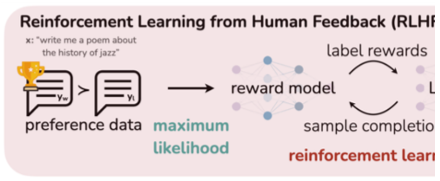
CSC 580: Artificial Intelligence II
Winter 2026

Homework #5: Reinforcement Learning Human Feedback (RLHF)

This assignment is to implement RLHF: an approach or training methodology that uses reinforcement learning and human provided preference data to align a model's behavior with human expectations.


Reinforcement Learning from Human Feedback (RLHF
x: "write me a poem about the history of jazz"
preference data
maximum likelihood
reward model
label rewards
sample completio
reinforcement lear

Note: You should do this assignment on platforms that provides GPU support (and high RAM ideally) because it involves intensive computation. The code provided will run on **Google CoLab** but should be fine on most other platforms.

Overview:

The assignment consists of two parts. You must do both of them.

- Part 1: Complete the given start-up code.
- Part 2: Make improvements to the code. You will use your own ideas to improve the performance (of four models) and carry out the experiments.

Part 1

- The start-up code "Simple_RLHF.ipynb" (posted on the **course Github site**) is a simple implementation of RL. It uses GPT-2 as the base model (for both Reward and Policy models, though for different tasks). The code essentially implements the basic "**three steps of RLHF implementation implements"** (shown on Slide 38 of lecture note #8):
  1. Supervised Fine Tuning (SFT) of Policy Model
  2. Reward Model Training (RMT)
  3. Reinforcement Learning of Policy Model (using the PPO algorithm)
- Your job is to fill in the places indicated with **"(*) TODO"**. There are 15 of them.
- After you complete the code, run the whole code from start to end, and ensure the outputs are displayed in the code.
- Report the results in the write-up as specified below (*).

Part 2

- In this part, you start with the completed code from Part 1 and make some changes to improve the performance of the models.
- To get started, you copy the code from Part 1 and save it in a different name. You make modifications in this file.

- In Part 2, you try a few things that would improve the results from Part 1 and observe the effects.
- You also write rigorous reasoning and thoughtful reflections and analysis on the experiment results in the write-up.

Initial comments:

- You may have noticed that the generated responses from all models (Base, SFT, Poor RM Policy, Good RM Policy) are not so great. They might be:
  - Incoherent or grammatically incorrect
  - Too short or cut off mid-sentence
  - Repetitive or nonsensical
  - Simply failing to generate any meaningful response
- THIS IS NOT A BUG -- it's a feature of the assignment!
- Your task is to diagnose WHY the performance is poor, and implement improvements and conduct experiments.
- Note that these experiments are open-ended -- there's no single correct answer. Your submission must show **well thought-out experimentation and rigorous analysis**.

TODO:

There are a total 4 tasks. Tasks 1 and 2 are meant to help you organize your thoughts for the experiments. Tasks 3 and 4 are the actual implementation and analysis of results.

- **Task 1: Diagnosis**
  - **Training Data Analysis**.
    - Examine the datasets used (SFT dataset: 5000 examples from Databricks Dolly-15k Preference dataset: 8000 examples from Anthropic HH-RLHF Training prompts: 500+ diverse prompts).
      Consider:
      - Is 5000 SFT examples enough?
      - What do production systems use?
      - How does data quality affect model performance?
      - Are the prompts appropriately formatted for GPT2?
    - Questions to answer:
      - Research how much data is typically used for SFT in production RLHF systems. How does our 5000 examples compare?
      - Look at some examples from the Databricks Dolly dataset. Are they well-suited for a small model like DistilGPT2?
      - The prompts end with colons (e.g., "Explain AI:"). Is this the best format for GPT-2 style models? What format were they originally trained with?
  - **Training Dynamics**
    - Analyze your **training curves** and write your answers in the write-up document.
      - Did the SFT loss decrease sufficiently?
      - How did the reward model accuracy evolve?

        - What do the PPO reward and KL curves tell you?
    - Questions to answer:
        - What was your final SFT loss? How does it compare to the initial loss?
        - Did the reward models reach high accuracy? If not, why might that be?
        - Looking at your PPO training curves, did the policy appear to be learning? What signs would indicate successful learning?

- **Task 2: Hypothesis Formation**
    - Based on your diagnosis, form hypotheses about the main causes of poor performance.
    - List **at least 5 hypotheses for potential causes**. For each cause, explain:
        - Why you think this might be a problem
        - How it would affect model outputs
        - Evidence from your experiments that supports this hypothesis
    - HINT: Example hypotheses:
        - "The model is too small to understand complex instructions because GPT2 was designed for simple text completion, not instruction following."
        - "The SFT training was insufficient because we only used 5000 examples and 200 steps, whereas instruction tuning typically requires more data."
        - "The generation parameters (temperature=0.7) cause too much randomness for a small model, leading to incoherent outputs."
    - Prioritize your hypotheses
        - Rank your hypotheses from most to least likely. For each, explain:
            - Why you ranked it this way
            - What experiment you could run to test it
            - What result would confirm or disprove it

- **Task 3: Improvement Experiments**
    - Choose **3 (three) of the following improvement options** to implement and test. For each:
        1. Implement the change in your code
        2. Run a small experiment (you can reuse trained models where possible)
        3. Document the results (including both quantitative metrics and qualitative examples)
        4. Analyze whether it helped and why
    - **Improvement Options:**
        - Option 1: **Better Generation Parameters**
            - Modify the generate_response function to use:
                - Lower temperature (try 0.3, 0.5, 0.7)
                - Top-k sampling (try k=20, 40, 50)
                - Top-p nucleus sampling (try p=0.8, 0.9, 0.95)
                - Repetition penalty (try 1.1, 1.2)
                - No repeat n-gram size (try 2, 3)
            - Observe: Which combination works best? Why?
        - Option 2: Prompt Engineering
            - Rewrite the training and test prompts to be more suitable for GPT-2:
                - Use continuation style instead of instruction style
                - Add examples (few-shot prompting)
                - Simplify language
                - Match GPT-2's training distribution
            - Example transformation:
                - Original: "Explain artificial intelligence:"
                - Transformed: "Artificial intelligence is a field that" Test both formats and compare.
        - Option 3: Increase SFT Training
            - Train the SFT model for more steps (try 500, 1000) or with more data.
                - Does performance improve with more training?
                - Is there a point of diminishing returns?
        - Option 4: Adjust PPO Hyperparameters
            - Experiment with:
                - KL weight (try 0.001, 0.01, 0.1)
                - Learning rate (try 1e-6, 5e-6, 1e-5)
                - Number of PPO episodes
        - Option 5: Reward Model Improvements
            - Try to improve the reward model:
                - Train for more steps
                - Use a different architecture (e.g., add a pooling layer)
                - Adjust the Bradley-Terry loss (e.g., add margin)
        - Option 6: Response Length Adjustment
            - GPT-2 struggles with long generations. Try:
                - Shorter responses (max_tokens=20, 30, 50)
                - Greedy decoding instead of sampling for evaluation
                - Early stopping at natural break points
        - Option 7: Ensemble Generation
            - Generate multiple responses and select the best one:
                - Generate 3-5 responses per prompt
                - Use reward model to score them
                - Select the highest scoring response
        - Option 8: Your Own Idea
            - Propose and implement your own improvement. Get instructor approval first.

- **Task 4: Reflection and Recommendations**
    - **Synthesis**
        - Based on all your experiments, write a comprehensive analysis:
            1. What was the PRIMARY cause of poor performance? Support with evidence.
            2. Which improvement was most effective? Why?
            3. What is the theoretical upper bound of performance with GPT2? Could it ever produce truly coherent instruction following?
            4. How would your improvement strategy change if you had access to more computational resources (e.g., a 7B parameter model)?
    - **Broader Implications**

- Connect your findings to real-world ML development:
    1. In industry, models often underperform initially. How would you systematically debug poor performance in a production system?
    2. What are the trade-offs between using a smaller, faster model vs. a larger, more capable model?
    3. How do the "last mile" problems you encountered (generation quality, prompt engineering) reflect real challenges in deploying LLMs?

Bonus: Extra Credit

Successfully implement and demonstrate a significant improvement in model performance. "Significant" means:

- Human evaluation shows clear improvement on at least 10 test prompts
- Quantitative metrics improve by >20%
- You can articulate WHY the improvement worked

Examples of bonus-worthy improvements:

- Implement a working version with GPT-2 medium (345M) despite memory constraints (hint: use gradient checkpointing or LoRA)
- Create a simple few-shot prompting system that actually works
- Implement a working inference-time intervention (e.g., sentiment steering)
- Train a reward model that *actually* prefers good responses over bad ones

---

Deliverables

Submit the following:

1. Shared link to the folder on your Google Drive account where you have your code and materials.
2. Write-up report.
    - **Minimum 2.5 pages** (in pdf of docx).
    - **Your name, student ID, course number and assignment number (HW#5)** at the top of the document. Failure to comply with this will result in significantly reduced grades.
    - **Collaborators** if you worked with other students or used **GenAI tools**. Claim any reference sites you consulted as well.

    - (*) Your comments and analysis on the overall system performance in Part 1, including comments on the outputs for (*) TODO 15 (Part 9: Evaluation using the HELD-OUT test prompts).
    - Your answers to Tasks 1, 2 and 4 in Part 2. To re-iterate, your submission must show **wwell thought-out experimentation and deep, rigorous analysis**./li>
    - Write your general reflections on the problem:
        - What you learned from this exercise.
        - How difficult you felt this exercise was.
        - Any particular difficulties you encountered.
        - How you would do/approach differently next time (if there was one).
        - and anything else.

Submissions

In the submission box 'HW#5'. Upload the write-up file. Type in your shared Google Drive folder link in the submission comment box.