Preferences vs Rewards

There are problems where providing *exact scalar rewards*

- is harder than *ranking* potential outputs.

For example

- I may prefer chocolate to vanilla
- but I can't quantify how much more

Technically

- rewards form a total order
    - a reward has a magnitude
    - *all* rewards can be compared and ordered

- preferences form a partial order

    - we can order *some* pairs of outputs

    - without providing a magnitude

    Good > Bad

    Big > Small

    Partial order:

    ```
    Good > Small ?   undefined
    ```

Problems related to aligning the *style* of an LLM's output is a case of preferences.

- multiple answers may be "correct"
- but one answer may be "preferred"

For example

Prompt: "How do I change a tire?"

- **Reply A:** An accurate step-by-step answer.
- **Reply B:** A brief, incomplete answer.

Both replies are "correct" but the first is subjectively better.

An example of *Preference Data* is a triple

$$(x, y^+, y^-)$$

- input x
- the preferred output y^+
- the non-preferred output y^-

The case for preferences

Scenario	Why Preference Data?	Typical Example
RLHF & LLM alignment	Human feedback easier as comparisons	Choosing better LLM output
Hard-to-define or subjective “success”	Preference judgments more reliable	Dialogue, safety, style
Biased or noisy scalar rewards	Preferences less affected by outliers	Creative tasks, open-ended
Interpretability needs	Preferences can include rationales	Transparent value alignment
DPO-style methods	Direct optimization over preferences	Pairwise/choice based loss

In this module, we explore Reinforcement Learning for Preference Data.

Example of a Preference Dataset

Here is a [link (https://huggingface.co/datasets/HuggingFaceH4/ultrafeedback_binarized)](https://huggingface.co/datasets/HuggingFaceH4/ultrafeedback_binarized) to the UltraFeedback dataset.

It is used to train an Assistant to be

- Helpful
    - answers the user's prompt; doesn't evade or decline
- Honest
    - gives a truthful answer

The methodology for constructing it is given [here (https://huggingface.co/datasets/HuggingFaceH4/ultrafeedback_binarized#dataset-card-for-ultrafeedback-binarized)](https://huggingface.co/datasets/HuggingFaceH4/ultrafeedback_binarized#dataset-card-for-ultrafeedback-binarized).

- The authors gather a number of prompts across multiple domains.
- An AI assistant is asked to proved multiple responses to a prompt
- A second AI assistant
    - critiques the response based on, e.g., the Helpfulness criteria
    - gives a numerical evaluation
- Which creates a ranking of the responses to a prompt

Synthetic Data in practice

The "binarized" dataset that we viewed

- selects the highest ranked answer as "Chosen"
- randomly selects the other responses as "Rejected"

Note the use of AI feedback rather than Human Feedback.

LLM Next Token prediction task and Reinforcement Learning

One motivation for Preference Methods

- is to post-train an LLM
- to exhibit desirable characteristics
- which is often expressed with Preference Data
    - Preferred output vs. Non-Preferred output

We translate the LLM "Next Token Prediction" task to an instance of Reinforcement Learning

- in order to be able to use Reinforcement Learning for post-training an LLM

The Language Modeling task, formally, is

- Predict the Next Token conditional on the previously generated tokens of the response
- producing output y given input x

For each output sequence $\y$ of tokens

- We can associate a *state* corresponding to *each prefix* of y
  - e.g, $\stateseq_= \y_{[0:-1]}$
- An action is
  - choosing a token from the Vocabulary as the next output
- The policy $\pi_\theta(\act|\state)$ is thus equivalent to

$$\pi_\theta(\y_| \y_{[0:-1]})$$

- the probability distribution of the next token, conditional on the prefix

Each sequence $\y$ becomes an episode/trajectory.

We can write the probability $\pi_\theta(\backslash\mathbf{y}|x)$

- of the *entire sequence*
- as the chained probabilities of each action given a state

This will be convenient

- in that we can compare the probabilities of
- a Preferred response y^+ to a Non-Preferred response $y^=$

rather than having to write the chained probability of each token in the sequences.

In [2]:

```
print("Done")
```

Done