Reinforcement Learning Fine-Tuning of Large Language Models for Domain-specific Customization to Auto-generate PFDs and P&IDs for novel Chemical Production Process

Sagar Srinivas

April 17, 2025

Limitations of Supervised Fine-Tuning (SFT)

- SFT adapts LLMs by learning from labeled input-output pairs using next-token log-likelihood loss.
- However, SFT cannot enforce nuanced domain-specific goals such as:
  - Long-term coherence and factual alignment.
    (e.g., In legal text, generating a consistent argument over multiple paragraphs.)
  - Adherence to domain-specific policies or constraints.
    (e.g., Medical models must avoid suggesting unverified treatments.)
  - Preference for concise or verbose reasoning depending on context.
    (e.g., Software Release Notes vs. Developer Documentation. Same event or fact (a software update) is described differently depending on the reader- RL can help by aligning outputs to contextual preferences — whether to be concise or elaborate)
- Outputs may be syntactically correct but semantically suboptimal for real-world domain needs.
  (e.g., A chemistry answer might look fluent but describe the wrong reaction conditions.)

- RL fine-tunes LLMs by treating them as policies $\pi_\theta(y|x)$ that generate full sequences.
- Instead of optimizing log-likelihood, we optimize expected utility:

$$\mathbb{E}_{y \sim \pi_\theta(\cdot|x)}[R(x, y)]$$

- $R(x, y)$ is a reward function capturing desired properties such as:
  - Task completion correctness (e.g., chemical process QA)
    (e.g., Correctly describing the heat exchanger role in ammonia synthesis.)
  - Preference alignment (via human feedback or a reward model)
    (e.g., Preferring answers that are easier to understand for high-school students.)
  - Domain-specific compliance (e.g., safety, legal constraints)
    (e.g., Never recommending toxic solvents in pharmaceutical synthesis.)
  - Response style adaptation (e.g., varying tone, formality, or verbosity)
    (e.g., Explaining a drug mechanism casually for patient education vs. formally in a research paper.)

1. **Policy Initialization:** Use a pretrained instruction-tuned model π_{θ_0} as the initial policy.
   (Start with a model already trained to follow instructions reasonably well.)
2. **Prompt Sampling:** Sample a batch of prompts x_i from domain-specific datasets.
   (e.g., Questions like "How is nitric acid produced?" from chemical engineering.)
3. **Response Generation:** Generate outputs $y_i \sim \pi_\theta(y|x_i)$.
   (The model attempts to answer these domain questions.)
4. **Reward Assignment:** Evaluate responses using:
   - Human-labeled preferences *(e.g., Annotators prefer Answer A over B.)*
   - Automated reward models *(e.g., Another LLM checks if the answer is factually correct.)*
   - Domain-specific constraints or verification tools *(e.g., Using a simulator to verify reaction steps.)*
5. **Policy Optimization:** Update π_θ using PPO, DPO, or GRPO to maximize reward.
   (Make the model learn to prefer high-quality answers that pass the reward test.)

- Directly aligns model behavior with domain-defined quality measures.
  (e.g., Optimizing to write safe and regulation-compliant chemical procedures.)
- Enables preference-driven learning where gold outputs are ambiguous or subjective.
  (e.g., Multiple ways to describe the same medical procedure.)
- Reduces hallucinations in sensitive domains like medicine or law.
  (e.g., Avoids fabricating case citations or nonexistent compounds.)
- Allows continuous improvement by integrating live feedback from deployed systems.
  (e.g., Learning from user upvotes in a legal advice platform.)
- Supports low-resource settings via reward models instead of labeled data.
  (e.g., No need for manual answers if we have a strong evaluator model.)

Use Cases: Chemical Process QA, Scientific Text Generation, Legal Drafting, Medical Report Summarization

LLM-Powered Framework for Automated PFD & P&ID Generation

- **Goal:** Automatically synthesize industrial-grade PFDs and P&IDs given a chemical name (e.g., `"Nitric Acid"`).
- **Input:** Natural language chemical identifier (e.g., `"Ammonia"`, `"Hydrogen Peroxide"`).
- **Framework:** A **meta-agent** orchestrates the task using instruction-tuned **LLMs** and tool-augmented sub-agents.
  - Subtasks include:
    - Retrieving synthesis pathways via web search and Wikipedia APIs.
    - Identifying major equipment and process steps (e.g., reactors, etc).
    - Inferring instrumentation and control logic for P&ID synthesis.
  - Sub-agents interface with external tools (e.g., DWSIM, Visual Paradigm) and chemical databases.
  - Outputs are rendered in structured textual formats or exported to simulation environments.
- **Output:**
  - **PFD:** High-level flow of materials and energy across unit operations.
  - **P&ID:** Detailed instrumentation—sensors, valves, controllers (e.g., LIC-101, TIC-203).
- **Applications:** Process design automation, DWSIM model bootstrapping, literature-to-process graph generation.

- **What is GRPO?**
  - A **reinforcement learning** (RL) algorithm designed to fine-tune **large language models** (LLMs).
  - Optimizes model behavior by ranking multiple generated responses for the same prompt.
- **Key Idea:**
  - Instead of using a separate value function, GRPO uses **group-based reward normalization**.
  - Within each group (prompt-response set), responses are scored and normalized to estimate their relative quality.
  - This allows for **advantage estimation** without a critic model.
- **Benefits:**
  - Simpler and more efficient than traditional policy optimization methods like **PPO (Proximal Policy Optimization)**.
  - Enables stable RL fine-tuning even with limited feedback data.
- **Applications:**
  - Used in training models like **DeepSeek-R1** for math reasoning, QA, and code generation tasks.

Composite Reward GRPO: Our Extension for PFD/P&ID Generation

- We extend standard GRPO to fine-tune a small-scale language model (SLM) for structured generation of process flow diagrams (PFDs) and piping and instrumentation diagrams (P&IDs).
- The model is treated as a stochastic policy $\pi_\theta(y \mid x)$, where x is a process synthesis query (e.g., `"Describe the PFD for nitric acid"`).
- For each prompt x, we sample a group of outputs $\mathcal{O}(x) = \{o_1, o_2, \ldots, o_G\}$ from the old policy $\pi_{\theta_{\text{old}}}$.
- We introduce a task-specific **composite reward**:

$$r(o_i, r_x) = 0.3\, r^{\text{rouge}} + 0.2\, r^{\text{length}} + 0.5\, r^{\text{LLM-domain}}$$

- Reward components:
  - $r^{\text{rouge}}(o_i, r_x)$: Surface-level semantic alignment with reference descriptions.
  - $r^{\text{length}}(o_i, r_x)$: Penalizes verbosity mismatch (under/over-detailed).
  - $r^{\text{LLM-domain}}(o_i, r_x)$: Judgment from a domain-aligned reward model that verifies process structure (e.g., unit operations, flow direction, safety logic).
- This domain-aware reward design guides the model to produce both technically accurate and well-structured PFD/P&ID descriptions.

Policy Optimization with Composite GRPO (Ours)

- After computing group-level rewards $\{r_1, \ldots, r_G\}$, we normalize them to obtain relative advantages:

$$\mu_x = \frac{1}{G}\sum_{i=1}^{G} r_i \qquad \sigma_x = \sqrt{\frac{1}{G}\sum_{i=1}^{G}(r_i - \mu_x)^2}$$

$$\widehat{A}_i = \frac{r_i - \mu_x}{\sigma_x}$$

- Each output $o_i = (o_{i,1}, \ldots, o_{i,T})$ represents a structured description of a PFD or P&ID.
- For each token t, compute the probability ratio between new and old policies:

$$r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} \mid x, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid x, o_{i,<t})}$$

- Our modified GRPO objective:

$$J_{\text{GRPO}}(\theta) = \mathbb{E}\left[\sum_{i=1}^{G}\sum_{t=1}^{|o_i|} \min\left(r_{i,t}(\theta)\widehat{A}_i, \text{clip}(r_{i,t}(\theta))\widehat{A}_i\right)\right] - \beta D_{\mathrm{KL}}(\pi_\theta \| \pi_{\text{ref}})$$

- This policy update promotes domain-aligned structure generation, with no critic model required.