Slide

Development and Working Principle of the Resistive Plate Counter (RPC)

Introduction (show lab photo)

- I' ll introcude development and working principle of the RPC.
- This photo is from our lab. On the right, you can see **four detector layers**.
- Our main goal is to **detect muons**.
- The idea is simple: we **track the muon' s path** as it enters and exits the detector.
- By computing the **intersection** of the incoming and outgoing tracks, we estimate **where the muon scattered**.
- Because **scattering probability grows with material density**, if we collect many such points, we get a kind of **"perspective" map** of the object we' re probing.

Slide

How an RPC Works

- Each RPC panel tells us **where** and **when** a muon passes.
- The panel is **hollow**, filled with **inert gas**, with **high voltage** between the top and bottom plates.
- As a muon goes through, the gas **ionizes** and triggers an **avalanche** of charges. The inert gas **keeps that avalanche under control**.
- The resulting **charged particles** are picked up by **our detector**, which lets us know **where** and **when** the muon passes.

Slide

My Work in the Lab

- I helped build a **larger-area detector** and **calibrated** it so it worked properly—basically a **larger** version of the original one.

- My advisor then gave me a challenge: we usually assume **single scattering**—so we take the **intersection** of the in/out tracks as *the* scattering point.
- In reality, a muon can **scatter twice or even three times**.
  It’s rarer, but it lowers resolution.
- I didn’t know much about AI at that time, but I was curious whether **AI could help with multiple scattering**, so I started learning—and that’s how I got into AI.

Divider

AI Research on Multimodal Models

Slide

My AI Research Journey

- After a summer of learning, I realized neural network had nothing to do with the **multiple-scattering** problem.
- But my interest in AI kept growing, so I started a project with a senior student.
- We focused on **reinforcement learning** and **test-time compute** to improve **image–prompt matching**.

Slide

How Our Model Generates Images (latent tokens)

- The model **doesn’t generate pixels directly**. It first generates **latent tokens**, then **decodes** them into an image.
- Here are 9 tokens
- **Start**: all tokens are **masked**.

- **Sampling**: for each position *(i, j)*, sample a token from **P(token$_{ij}$ | masks, prompt, pos$_{ij}$)**.
- This is **fully parallel**.
- **Iterate**: after the generation, **fix high-confidence tokens**, keep the rest masked, and **repeat**.
- **Finish** when no masks remain, then **decode tokens → pixels** to get the final image.

Slide

Using RL for Optimization (DPO)

- At the end of generation, we score **how well the image matches the prompt**—that's our **match score**.

Slide

How to score a Candidate

- We feed a candidate image to another model and ask: **"Does this image match the prompt?"**
- Instead of a hard yes/no, we look at the **probability of "Yes"** vs. "No".
- Our **match score** = P("Yes") − P("No").

- We use **DPO (Direct Policy Optimization)** with **good vs. bad pairs**: increase the probability of **good samples**, decrease that of **bad samples**.
- With enough data, this will **improve the alignment** during training.

Slide

Test-Time Compute: ORM, PRM, and PARM

- **ORM (Outcome Reward Model):** "best-of-N"—generate N images and pick the best. Works well, but **expensive**.
- **PRM (Progressive Reward Model):** make **step-by-step selections** during generation. But it struggles because **early images are too blurry**.
- **PARM:** our idea. We check **intermediate images**; if a branch shows **potential to match the prompt**, we keep it; weak branches are **pruned early**. That **cuts unnecessary compute** while maintaining quality.

Slide

Conclusion and Results (for this part)

- We combined **training-time DPO** with **test-time methods** (ORM/PRM and our **PARM**).
- Together, they **improved prompt–image alignment**.

Divider

Overcoming Challenges with GRPO

Slide

From PPO/DPO to GRPO

- After that work, DeepSeek drew a lot of attention. Over winter break I compared **PPO, DPO, and GRPO**, then started independent work with a professor at **UIUC**.
- I started with **replicating GRPO** and understanding its behavior. Soon, I found a problem.

Slide

Sampling vs. Reward Distributions

- Given a **prompt/condition**, the model outputs a **probability distribution over actions**. We **sample** from that distribution.
- The **reward distribution** tells us which action would **score well**.

Slide

What GRPO Does

- For a given prompt, GRPO **samples many sequences** (orange dots).
- Each sample gets a **reward**.
- We compute **relative advantages** within the group: increase the prob of sequence with **positive** advantage, decrease those with **negative**.
- Over time, the sampling distribution **shifts toward** the reward distribution.

Slide

The Main Issue

- If the **sampling distribution** and the **reward distribution don't overlap** at all, then **all sampled rewards are zero**.
- With **no relative advantages**, GRPO **can't make progress**.

Slide

Two Fixes I Proposed

1. **Sampling Probability Diffusion**
    - If a group's rewards are **all zero**, we set **every sample's advantage negative**.
    - That means we decrease all samples' porb.
    - This will **flatten** the current probability peak and can **push** sampling **toward** the reward peak.
2. **Embed a Seed in the Reward Distribution**

    - After normal sampling, **replace the final free sampled answer with ground truth**, while **keeping the reasoning trace**.
    - Then compute **P(ground_truth | reasoning_trace)** and **use it as the reward**.
    - Intuition: a **better reasoning trace** should make the **ground truth more probable**.
    - Because the answer tokens are now ground truth, their loss is **cross-entropy** (not policy gradient).

Slide

New Training Process

- This handles the **zero-reward** case.
- Each time we backprop with **cross-entropy on answer tokens**, we're **teaching the answer**. That is **planting a small probability peak** inside the reward region.
- Training recipe:
    - **Per epoch**, for groups with **zero rewards**, use **our loss**.
    - For the rest, **stick to the original loss**.
    - Over epochs, the hardest subset—the "**iceberg**" the original GRPO never touched—**melts away**.

Divider

Crystal Structure Reasoning-Path

Slide

Setup (move to CUHK)

- In the summer I planned to continue at UIUC, but cluster issues forced me to pause. I then joined a professor at **CUHK** with a new direction.

- Task: **feed a crystal structure in**, and have the model **output a reasoning trace** for generating it.
- The reasoning process is something like this: "Based on symmetry, blah blah blah..."
- Then we give the **reasoning trace** to another model that **reconstructs the structure**
- The entire process can be viewed as **encode → language-space → decode**.
- If this works, we can examine whether the **trace** conflicts with **modern physics**—maybe even hint at **new physics**.

Slide

Challenge and Approach

- At the early stage, We let the model generate structures **freely**, it may produce **wrong atom counts or types**, etc.
- Using original GRPO here is hard because **designing a good reward function** is tricky.
- The method I developed at UIUC fits perfectly: directly use **P(crystal structure | reasoning trace)** as the reward.

Slide

Experiment—and Why It Failed

- AI didn't behave like human reasoning.
- The model **didn't avoid leaking the answer** in the reasoning trace.
- If the trace contains information about the correct answer, then **P(crystal|trace)** becomes **very high**, and the model gets rewarded without honest reasoning.

Slide

Conclusion for This Part

- Over time, the model started **embedding answers into the trace**—often in **natural-language**.
- Example: coordinates **(0.5, 0.5, 0.5)** described as **“the atom is at the center of the lattice.”**
- So what we saw wasn’t discovery of new physics, but a tendency to **embed the answer into the reasoning**.
- The leakage problem led me to conclude that this idea had failed.