Training Gemma 2 2B to Reason with Tunix

Introduction

This project trains Google's Gemma 2 2B model to produce structured reasoning traces before answering questions. Using Tunix, Google's JAX-native library for LLM post-training, we implement a three-stage training pipeline: Supervised Fine-Tuning (SFT), Direct Preference Optimization (DPO), and Group Relative Policy Optimization (GRPO). The goal is to teach the model to "show its work" in a consistent XML format:

```
 <reasoning>step-by-step thought process</reasoning>
<answer>final answer</answer>
```

Training Strategy

Three-Stage Pipeline

Our approach progressively refines the model through three complementary training methods:

Stage 1: Supervised Fine-Tuning (SFT) — We first teach the model the target output format by training on high-quality examples. This establishes the foundation for structured reasoning.

Stage 2: Direct Preference Optimization (DPO) — Next, we refine the model's preferences by training on pairs of chosen (good) and rejected (bad) responses. This helps the model distinguish between high-quality and low-quality reasoning and responses.

Stage 3: Group Relative Policy Optimization (GRPO) — Finally, we use reinforcement learning with programmatic reward functions to improve answer accuracy and reasoning. GRPO generates multiple responses per prompt and uses reward signals to reinforce correct answers.

Compute Allocation

We allocated our 9-hour Kaggle TPU session as follows:

Stage	Time	Size (trained records)
SFT	22 mins	~3000
DPO	16 mins	~5800
GRPO (Math)	56 mins	~4500
GRPO (Extra)	43 mins	~2500

Key Techniques

- **LoRA (Low-Rank Adaptation):** We use rank-64 LoRA adapters targeting attention layers (Q, KV, attention output) and MLP layers (gate, up, down projections). This enables efficient fine-tuning while preserving base model knowledge.
- **Mesh Configuration:** We used (8,1) FSDP for SFT/DPO to maximize memory efficiency and throughput for forward/backward passes, achieving an effective batch size of 32. For GRPO, we switched to (1,4) tensor parallelism to optimize per-token generation speed and reduce KV-cache pressure during rollout-heavy workloads with multiple generations per prompt.
- **Learning Rate Scheduling:** We use warmup (5-10% of training) followed by cosine decay, starting from 2e-5 for SFT and 3e-6 for DPO/GRPO.

Dataset Creation

SFT Dataset

Source: `kaggle.com/datasets/vennasan/sft-gemini-gen-data`

We constructed the SFT dataset by curating a diverse mix of public instruction datasets spanning math, coding, writing, summarization, puzzles, first-principles science, and commonsense reasoning.

Curation Process:

- **Domain Weighting:** Sampling weights were carefully balanced to emphasize skills that benefit most from supervised learning, particularly instruction-following and creative writing
- **Content Diversity:** Math samples included both standard and harder, less-verifiable problems; coding samples ensured broad language coverage (Python,

Java, SQL, etc.)
- **Format Standardization:** All samples were processed through Gemini-2.5-Flash to generate high quality and consistent XML-formatted responses with explicit `<reasoning>` and `<answer>` tags
- **Quality Control:** Rigorous filtering removed malformed outputs, duplicates, and low-quality examples
- **Evaluation Split:** A small held-out slice from each domain was reserved for regression testing during training

This approach ensured the model learned both the target output format and high-quality reasoning patterns across diverse problem types.

DPO Dataset

Source: `kaggle.com/datasets/sandeeplleb/dpo-final`

We constructed the DPO dataset through a two-round process focused on non-verifiable domains (writing, summarization, creative tasks):

Round 1: Multi-Model Generation & Ranking

- Sampled prompts from public datasets across writing, summarization, and creative domains
- Generated responses using three models: Gemma-2-2B-IT, Gemma-3-4B-IT, and Gemini-2.0-Flash
- Used Gemini-2.5-Flash to rank all responses based on reasoning quality and answer quality
- Programmatically constructed preference pairs:
  - **Chosen:** High-quality responses with proper XML formatting and strong Gemini rankings
  - **Rejected:** Poor formatting or low-quality responses with weak Gemini rankings

Round 2: Targeted Refinement

- Trained an offline Gemma-2-2B-IT model on the Round 1 dataset
- Generated responses on additional unseen prompts to identify weaknesses
- Sampled failure cases where the model struggled with formatting or response quality
- Added these targeted examples to create the final DPO dataset

This iterative approach ensured the DPO dataset emphasized the model's specific weaknesses in formatting consistency and output quality.

GRPO Datasets

Math Domain:

GSM8K benchmark

Extra Domains: `kaggle.com/datasets/sandeeplleb/grpo2-final`

We built this domain from verifiable publicly available datasets from domains such as puzzles, first principles and common sense.

Reward Function Design

For GRPO, we designed a composition of reward functions:

Math Rewards

- **Format Exact (3.0 pts):** Full XML structure with reasoning and answer tags
- **Format Approximate (2.0 pts):** Partial tag presence
- **Answer Correct (3.0 pts):** Exact match, substring match, or within 10% numeric tolerance
- **Numeric Backup (1.5 pts):** Fallback numeric extraction and comparison

Multi-Domain Rewards

- **Format (0.4):** Perfect XML structure compliance
- **Accuracy (0.4):** Domain-specific answer verification
- **Reasoning Quality (0.1):** Length, structure, logical flow, convergence to answer
- **Bonus (0.1):** Perfect format + accuracy + good reasoning

This multi-component reward design encourages both correct answers and well-structured reasoning traces.

Prompt Template

All training and inference uses a consistent prompt format:

```
 <start_of_turn>user
You are a meticulous AI expert. Your task is to solve a problem by first showing your work and then stating the final answer.

INSTRUCTIONS:
1. **Reasoning:** Break down the problem and explain your thought process. Enclose this entire thought process in between <reasoning
2. **Answer:** State only the final, conclusive answer. Enclose it in between <answer> and </answer>.

TASK:
'''
{question}
'''
<end_of_turn>
<start_of_turn>model
```

Results

The universal test dataset size is 478 questions which are made up of 10% of unseen data across all training domains.

Overall performance lift:

- Correct formatting increased by ~86% (51% to 95%)
- Reasoning quality increased by ~19% (from 5.6 to 6.7 on a scale of 1-10)
- Answer quality increased by ~18% (from 5.4 to 6.4 scale of 1-10)

Metric	Base	Post-SFT	Post-DPO	Post-GRPO
Overall (1–20)	11.01	12.58	12.48	13.14
Format PASS %	50.6%	89.5%	76.8%	94.8%
Reasoning (1–10)	5.59	6.43	6.41	6.75
Answer (1–10)	5.41	6.15	6.07	6.40

Figure: Performance after each training step.

Ablation Studies

1. Finding the ideal mesh configuration for each algorithm.
2. Experimented multiple rounds to determine the correct DPO weightage to ensure that formatting tags and high quality answers were prioritized.
3. Experimented different micro batch sizes across different algorithms to determine the best speed v.s. performance trade-off.

Challenges and Learnings

What I Learned

- **Data quality over quantity matters for SFT:** Using Gemini-generated reasoning traces with consistent XML formatting established a strong foundation. The 3,025 high-quality samples were more effective than a larger noisy dataset.
- **Mesh configuration significantly impacts training efficiency:** Switching from (8,1) FSDP for SFT/DPO to (1,4) tensor parallelism for GRPO was crucial—GRPO's rollout-heavy workload benefits from TP's better per-token generation efficiency and reduced KV-cache pressure.
- **Multi-component reward functions are essential for GRPO:** The importance of having granular reward functions for each behaviour we want to encourage is crucial when logging the reward function performance. If there was only one reward function with all the desired behaviours in it, then we wouldn't know which rewards the model is calibrating towards.
- **Progressive refinement works:** Each stage built meaningfully on the previous—SFT taught format (51%→90% format compliance), DPO refined preferences, and GRPO pushed accuracy further (final 95% format, +19% reasoning quality).

Challenges Faced

- **Memory constraints required creative solutions:** The Flax 0.12.0 compatibility patch for qwix was necessary to avoid crashes. Careful cleanup between stages (deleting trainers, clearing JAX caches) was essential to avoid OOM errors.
- **Cross-mesh checkpoint loading:** Loading DPO checkpoints (saved on 8,1 mesh) into GRPO models (on 1,4 mesh) required implementing a `force_reshard_model()` function to ensure consistent sharding.
- **Truncation management:** Ensuring prompts fit within MAX_PROMPT_LENGTH (256 for GRPO) while preserving question integrity required filtering ~500 samples from the GSM8K dataset.

Suggestions for the Hackathon

- **TPU queue time:** The wait times to get access to TPUs was very long throughout December and January. Allowing for more allocated resources would ensure a better developer experience.

Conclusion

This project demonstrates a practical approach to training reasoning models using Tunix. The three-stage pipeline (SFT → DPO → GRPO) progressively refines the model's ability to produce structured, accurate reasoning traces. By combining supervised learning, preference optimization, and reinforcement learning, we achieve a model that not only answers questions correctly but also explains its thought process in a consistent format.

Final Model: `sandeeplleb/tunix-gemma2-2b-grpo-final`
