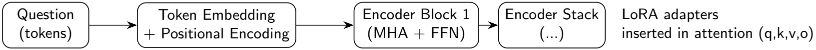
Hybrid DPO for Reasoning: Forward + Backward Preference Signal

Murtaza Nikzad

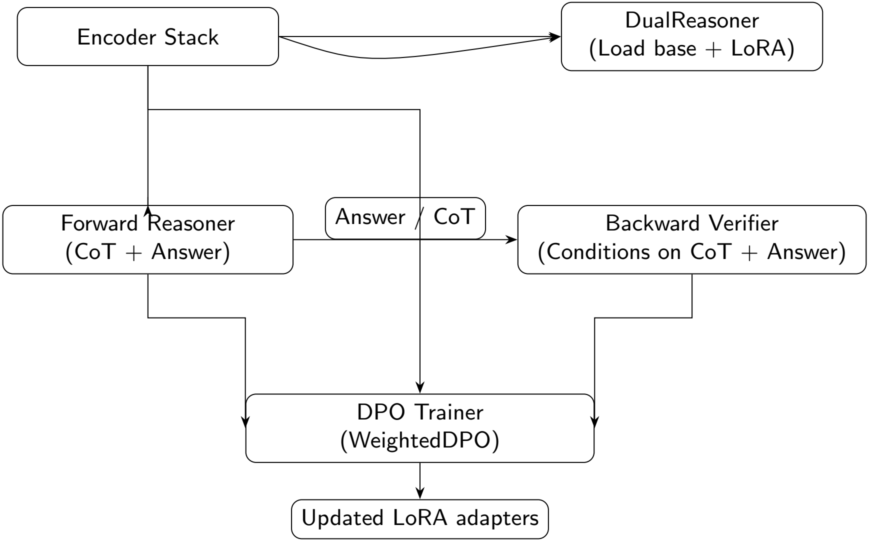
CS Research

December 14, 2025

Architecture — Encoder



Architecture — Reasoners & DPO



Pipeline Overview

1. Bootstrap paired dataset (forward traces, backward verification traces, gold answers) using the provided bootstrapping script.
2. Train DPO variants: baseline, forward-only, backward-only, and hybrid using the repository training scripts and configs.
3. Save LoRA adapters to the `outputs/runs/` directory and run the evaluation pipeline to produce `outputs/evals/` artifacts.
4. Extract per-example forward/backward traces and compare variants with the comparison utility.

Data and Bootstrapping

- Dataset derived from processed prompts under `data/processed/` (forward and backward JSONL).
- Bootstrapping pairs constructs (question, forward trace, backward trace, gold) per example.
- Example size: experiments reported on GSM8K subset (100-example trace comparison; full eval files in `outputs/evals/`).

Training: WeightedDPOTrainer

- DPO implemented with per-example weights (forward_weight, backward_weight) via `WeightedDPOTrainer`.
- Hybrid config uses forward_weight=0.6, backward_weight=0.4 (see `configs/dpo_hybrid.yaml`).
- Training uses LoRA adapter checkpoints to keep base model frozen.

Inference: DualReasoner

- `DualReasoner` generates forward trace and final answer, then runs a backward verifier conditioned on forward answer.
- Extractors (`extract_final_answer`, `extract_verification`) compute final outputs and PASS/FAIL.
- Supports loading base model + merged LoRA adapter for evaluation.

Experimental Setup

- Base model: Llama-3.1-8B (local checkpoint under models/).
- LoRA adapters: saved per-experiment under outputs/runs/ and loaded/merged for evaluation.
- Evaluations: JSON metrics and trace CSVs saved in outputs/evals/ (see repo outputs for full results).
- Reproduction commands are provided in the appendix slide.

Quantitative Results (Selected)

Model	Accuracy	Acknowledgement Rate
Baseline (orig)	0.80	0.60
GSM8K adapter	0.50	1.00
Hybrid DPO	0.81	0.947

Table: Metrics from `outputs/evals/*.json` (summary).

- Note: numbers above are taken from saved eval JSONs; per-sample tracing of 100 examples shows baseline 80/100 vs hybrid 83/100 (see appendix examples).

Per-example Trace Analysis (Representative)

- We saved 100 sample comparisons at `outputs/evals/gsm8k_samples_compare.csv`.
- Representative differing example (index 12):
  - Question: arithmetic reasoning; gold: 42
  - Baseline forward answer: 38 (verification: FAIL)
  - Hybrid forward answer: 42 (verification: PASS)
- 10 CSV examples exported to `outputs/evals/gsm8k_examples.csv` for inspection.

Discussion

- Hybrid DPO shows modest accuracy gains on the tested evaluation slices (0.80 -¿ 0.81 overall; 80/100 -¿ 83/100 in trace sample).
- Stronger verification signal increases model's ability to flag mistakes (acknowledgement changes), but evaluation definitions vary across scripts/heuristics.
- Per-example analysis shows hybrid helps on some error types (algebraic simplification, multi-step arithmetic) while leaving others unchanged.

Limitations

- Small-scale trace evaluation (100 examples) — not a full statistical analysis.
- Heuristics for extracting final answers and verification may mismatch gold formatting; need robust parsing.
- Training hyperparameters (LoRA rank, DPO weights) were not exhaustively tuned.
- Potential overfitting to verifier-style prompts; domain generalization untested.

Next Steps (Recommended)

1. Run full evaluation across entire GSM8K with bootstrap CI (bootstrap resampling) to assess significance.
2. Ablation sweep over `forward_weight` and `backward_weight` grid (e.g., 0.2 increments) to map trade-offs.
3. Improve answer extraction heuristics; add human-labeled verification subset to measure verifier accuracy.
4. Evaluate on out-of-distribution reasoning tasks to test generalization.
5. Explore verifier-only fine-tuning vs joint hybrid to compare approaches.

Repro: Key Commands

- Bootstrap pairs: `python scripts/bootstrap_pairs.py --out data/processed/...`
- Train (example): `python scripts/train_dpo.py --config configs/dpo_hybrid.yaml`
- Evaluate: `python scripts/eval_reasoning.py --config configs/eval_gsm8k.yaml`
- Compare traces: `python scripts/compare_traces.py --config-a configs/eval_baseline.yaml --config-b configs/dpo_hybrid.yaml --n 100`

Appendix: Files to Inspect

- Code: src/reasoning_lab/inference/dual_reasoner.py, src/reasoning_lab/training/weighted_dpo_trainer.py
- Configs: configs/dpo_hybrid.yaml, configs/dpo_forward_only.yaml, configs/eval_gsm8k.yaml
- Eval outputs: outputs/evals/eval_gsm8k_hybrid.json, outputs/evals/gsm8k_samples_compare.csv, outputs/evals/gsm8k_examples.csv
- Slides & paper: papers/hybrid_dpo_acm.tex, papers/hybrid_dpo_slides.tex, papers/hybrid_dpo_acm.pdf

Thank you

Questions? Discussion points for advisor:

- Prioritize CI vs more examples?
- Trade-off tuning strategy for verifier weight?
- Human-label verification subset size and selection?