Clase 150 — DPO y RLHF: alineamiento de LLMs

Parte: 2 — Deep Learning · Fuente: Ouyang et al. (2022) InstructGPT/RLHF + Rafailov et al. (2023) DPO + papers IPO/KTO/ORPO. Duración estimada: 95 min.

Clase 150 — DPO y RLHF: alineamiento de LLMs

> *Parte: 2 — Deep Learning · Fuente: Ouyang et al. (2022) InstructGPT/RLHF + Rafailov et al. (2023) DPO + papers IPO/KTO/ORPO. Duración estimada: 95 min.*

Objetivo

Alinear LLMs con preferencias humanas (helpful, harmless, honest). Cubrir RLHF clásico (SFT → Reward Model → PPO, complejo) y DPO (Direct Preference Optimization, moderno y simple). Conocer variantes 2023-2024: IPO, KTO, ORPO.

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Explicar el pipeline RLHF de 3 etapas (SFT, RM, PPO).
- Aplicar DPO con trl.DPOTrainer sobre un dataset de preferencias.
- Diferenciar DPO (single-step) de KTO (no requiere pairs) y ORPO (combina SFT + alineamiento).
- Crear un dataset de preferencias: (prompt, chosen, rejected).
- Evaluar alineamiento con MT-Bench, AlpacaEval, o LLM-as-judge.

Temas

- Por qué alineamiento: LLM pretrained → genera pero no sigue instrucciones bien ni evita harm.
- SFT (Supervised Fine-Tuning): instruction tuning.
- Reward Model: regression entrenada con human preferences.
- PPO: optimizar el LLM contra el RM.
- DPO: derivación matemática que elimina el RM.
- IPO: variante con identity link, más estable.
- KTO: solo chosen o rejected (no necesita pairs).
- ORPO: alineamiento desde SFT directamente.

Definiciones y características

- SFT: training supervisado con (prompt, response_humana).
- Reward Model: predicen r(prompt, response).
- Bradley-Terry: modelo probabilístico P(A > B) = σ(r(A) - r(B)).
- DPO loss: -log σ(β·(log π(chosen)/π_ref(chosen) - log π(rejected)/π_ref(rejected))).
- β (DPO): control del KL respecto al ref model. 0.1-0.5 típico.
- Reference model: copia frozen del SFT, usada para regularizar.

Dataset / recursos

- Anthropic HH-RLHF, UltraFeedback, Argilla DPO datasets.
- Modelo base: SFT propio (clase 128a) o Mistral 7B Instruct.
- Librerías: transformers, trl, peft, bitsandbytes.

Ejercicios

1. Dataset de preferencias: cargar Anthropic/hh-rlhf. Inspeccionar chosen y rejected.
2. DPO con TRL: DPOTrainer(model, ref_model, ...) con LoRA encima. Train 1 época.
3. Eval pre/post: generar respuestas a 20 prompts antes y después; comparar manualmente.
4. β sensitivity: probar β {0.1, 0.3, 1.0}. β alto → menos cambio; β bajo → más agresivo.
5. KTO: dataset con solo chosen (no pairs). Aplicar KTOTrainer.

Homework verificable

DPO sobre un dominio propio:

1. Dataset de 200-500 pares (puede ser sintético con LLM como judge).
2. Base: modelo SFT propio (de 128a).
3. DPO con LoRA, β=0.1.
4. Comparar 20 respuestas pre/post; evaluar con LLM-as-judge (e.g., Claude).

Criterio de aceptación: ≥ 60 % de los outputs post-DPO son juzgados mejores que pre.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
DPO degrada calidad	β muy bajo o demasiadas épocas. Fix: β=0.1
Reward hacking en RLHF	Modelo aprende a hacer trampas al RM. Fix:
ref_model consume mucha VRAM	Copia frozen del modelo. Fix: usar ref_mod
Dataset de pairs muy ruidoso	Annotators inconsistentes. Fix: filtering,
Resultados malos sin SFT previo	DPO asume modelo razonable. Fix: SFT prime

Preguntas frecuentes

DPO o RLHF clásico?

DPO por default 2024+ — más simple, casi igual calidad. RLHF si tenés team y reward model bueno (e.g., GPT-4 / Claude para producción).

¿IPO, KTO, ORPO cuál?

DPO sigue siendo default. IPO si DPO inestable. KTO si solo tenés chosen. ORPO combina SFT+pref → más eficiente, en alza.

¿Dataset de cuántos pairs?

10k-50k para alineamiento serio. 500-2000 para experimentos.

Evaluación cómo?

LLM-as-judge (GPT-4/Claude evalúa pares), MT-Bench, AlpacaEval. Human eval para gold standard.

¿DPO en LLMs > 70B?

Sí, con FSDP / DeepSpeed. Trabajo "expensive" pero factible.

Referencias

- Ouyang et al. (2022), Training language models to follow instructions with human feedback (InstructGPT), NeurIPS.
- Rafailov et al. (2023), Direct Preference Optimization, NeurIPS.
- Ethayarajh et al. (2024), KTO, NeurIPS.
- Hong et al. (2024), ORPO.
- TRL docs.

Siguiente clase

Clase 151 — vLLM y TGI: serving de LLMs en producción

Apéndice: notebook (primer bloque)

Implementamos DPO loss desde scratch con dataset sintético de preferencias. RLHF tradicional (SFT → RM → PPO) se explica conceptualmente.

```
import numpy as np
rng = np.random.default_rng(42)

# Dataset sintético: 100 pares (prompt, chosen, rejected) como embeddings
n_pairs = 100
d_emb = 32

# Generamos "prompts" y respuestas: chosen siempre tiene un offset hacia un "good direction"
good_dir = rng.standard_normal(d_emb)
good_dir /= np.linalg.norm(good_dir)

prompts = rng.standard_normal((n_pairs, d_emb))
chosen = rng.standard_normal((n_pairs, d_emb)) + 1.5 * good_dir
rejected = rng.standard_normal((n_pairs, d_emb)) - 0.5 * good_dir

print(f'pairs: {n_pairs}, dim={d_emb}')
print(f'chosen·good = {(chosen @ good_dir).mean():.3f} (alto)')
print(f'rejected·good = {(rejected @ good_dir).mean():.3f} (bajo)')
```

Archivos complementarios

- notebook.ipynb