Clase 155 — LLM Evaluation: MMLU, MT-Bench, LLM-as-judge, evals propios

Parte: 2 — Deep Learning · Fuente: Hendrycks et al. (2021) MMLU + Zheng et al. (2023) MT-Bench + LMSys Arena. Duración estimada: 85 min.

Clase 155 — LLM Evaluation: MMLU, MT-Bench, LLM-as-judge, evals propios

> *Parte: 2 — Deep Learning · Fuente: Hendrycks et al. (2021) MMLU + Zheng et al. (2023) MT-Bench + LMSys Arena. Duración estimada: 85 min.*

Objetivo

Evaluar LLMs (propios o terceros) con rigor: benchmarks estándar (MMLU, HumanEval, GSM8K, MT-Bench, LMSys Arena), LLM-as-judge para casos open-ended, y evals propios específicos al dominio. Reconocer las trampas (data contamination, reward hacking, leaderboard hacking).

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Correr MMLU con lm-evaluation-harness sobre un modelo propio.
- Implementar LLM-as-judge con (prompt, response_A, response_B) → "cuál es mejor".
- Diseñar evals custom: cobertura por tema, casos edge, regresiones.
- Diferenciar classification metrics (accuracy on MCQs) de generation metrics (BLEU, ROUGE, BERTScore, LLM-judge).
- Reconocer data contamination (test set en pretraining) y leaderboard hacking.

Temas

- MMLU: 57 dominios, multiple choice. Standard 2020-2023.
- HumanEval: 164 problemas Python codegen.
- GSM8K: math word problems.
- MT-Bench: 80 multi-turn questions evaluadas por GPT-4 judge.
- LMSys Arena: head-to-head humanos. Standard moderno (ELO ranking).
- LLM-as-judge: usar GPT-4/Claude como evaluador.
- Custom evals: críticos para producción.

Definiciones y características

- MMLU: 15k MCQs, 57 categorías. Score 0-100.
- MT-Bench: 80 prompts multi-turno; GPT-4 score 1-10 cada respuesta.
- LMSys Arena: humanos votan A vs B; ELO global.
- Pass@k: codegen — % de problemas resueltos en k intentos.
- LLM-as-judge: criterios estructurados, comparison pairs.
- Data contamination: el modelo memorizó el test → metrics infladas.

Dataset / recursos

- HuggingFace: lm-eval-harness, HELM, lighteval.

- Modelo a evaluar: cualquier LLM open o API.
- Librerías: lm-evaluation-harness, inspect_ai, promptfoo.

Ejercicios

1. MMLU con lm-eval-harness: lm_eval --model hf --model_args pretrained=mistralai/Mistral-7B-v0.1 --tasks mmlu --num_fewshot 5. Reportar score.
2. HumanEval: code generation, pass@1.
3. MT-Bench: usar GPT-4 / Claude como judge. Reportar score promedio.
4. LLM-as-judge propio: 20 pairs (model_A vs model_B); judge devuelve A/B/tie + reasoning.
5. Custom eval: 50 prompts específicos a tu use case + criterios de aceptación.

Homework verificable

Evaluar 2 modelos (e.g., Mistral 7B vs Llama 3 8B) en una tarea propia:

1. 30 prompts específicos al dominio.
2. Generar respuestas con ambos.
3. LLM-as-judge (Claude o GPT-4) comparando.
4. Reportar win rate de cada uno.

Criterio de aceptación: judge entrega resultado consistente (inter-rater agreement > 0.7 entre 2 ejecuciones).

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
MMLU score muy alto sospechoso	Posible data contamination. Fix: verificar
LLM-as-judge sesgado por longitud	Tiende a preferir respuestas largas. Fix:
LLM-judge sesgado por orden A/B	Si A siempre se evalúa primero, sesgo. Fix
Eval gameable	Optimizar en el test → degrada producción.
GPU OOM con lm-eval	Modelo grande. Fix: --batch_size auto:1 y

Preguntas frecuentes

MMLU sigue siendo válido en 2026?

Sí pero saturado — top modelos > 88 %. Mejores: MMLU-Pro, GPQA, BBH, MATH.

LMSys Arena reliable?

Sí, gold standard humano. Caro (necesita usuarios). Para producción usar como ground truth.

LLM-as-judge confiable?

GPT-4 / Claude como judge correlatean ~85 % con humanos en MT-Bench. Bias conocidos (length, position). Mitigar con prompts cuidadosos.

Evals para agentes?

SWE-Bench (coding), τ-Bench (tool use), AgentBench. Custom para tu workflow.

Eval continuo en producción?

Sample logs → LLM-as-judge daily → alertar si win-rate baja vs baseline.

Referencias

- Hendrycks et al. (2021), MMLU, ICLR.
- Zheng et al. (2023), Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena, NeurIPS.
- LM Evaluation Harness.
- LMSys Arena.
- Inspect AI — UK AISI eval framework.

Siguiente clase

Clase 156 — Autoencoders: undercomplete, stacked, denoising, sparse

Apéndice: notebook (primer bloque)

Simulamos MMLU, MT-Bench y un LLM-as-judge sintético. Discutimos biases comunes del judge.

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

Archivos complementarios

- notebook.ipynb