Clase 171 — Aceleración con GPU

Parte: 2 — Deep Learning · Fuente: Géron, cap. 19 § Using GPUs to Speed Up Computations. Duración estimada: 60 min.

Clase 171 — Aceleración con GPU

> *Parte: 2 — Deep Learning · Fuente: Géron, cap. 19 § Using GPUs to Speed Up Computations. Duración estimada: 60 min.*

Objetivo

Configurar GPU para DL: drivers, CUDA, cuDNN, verificación. Conocer mixed precision (bfloat16/float16) que duplica throughput en GPUs modernas (Ampere/Hopper). Profilear con TensorBoard Profiler para identificar bottlenecks (data loading vs compute vs sync).

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Verificar GPU: tf.config.list_physical_devices('GPU'), torch.cuda.is_available().
- Activar mixed precision: keras.mixed_precision.set_global_policy('mixed_float16') (Volta+) o 'mixed_bfloat16' (Ampere+).
- Limitar memoria GPU: set_memory_growth(True) para no consumir toda al inicio.
- Multi-GPU básico con tf.distribute.MirroredStrategy (clase 144 profundiza).
- Profilear con TensorBoard Profiler.

Temas

- CUDA toolkit + cuDNN versions matching.
- nvidia-smi para monitor.
- Mixed precision: float16 (Volta+, 16 GB max) vs bfloat16 (Ampere+, mismo exponente que float32, más estable).
- Memory growth vs allocated all.
- Profiling con TensorBoard.
- GPUs típicos en 2026: H100, H200 (server); RTX 5090 (consumer).

Definiciones y características

- CUDA: API de NVIDIA para GPU compute.
- cuDNN: librería de primitives DL (Conv, RNN, MatMul).
- Mixed precision: usar float16/bfloat16 para forward+backward, float32 para weights master copy.
- mixed_float16: requiere loss scaling para evitar underflow.
- mixed_bfloat16: mismo rango que float32, no necesita scaling.
- TPU: ASIC de Google. Diferente API (XLA).

Dataset / recursos

- Modelo + dataset cualquier de las clases previas.
- Librerías: tensorflow, opcional nvidia-smi.

Ejercicios

1. GPU check: imprimir tf.config.list_physical_devices('GPU'), tf.config.list_logical_devices('GPU'), nvidia-smi.
2. Memory growth: tf.config.experimental.set_memory_growth(gpu, True) para evitar reservar 100 % al inicio.
3. Mixed precision: keras.mixed_precision.set_global_policy('mixed_float16'). Re-entrenar modelo. Verificar speedup (1.5-2× en V100; 2-3× en A100/H100).
4. Profile: keras.callbacks.TensorBoard(log_dir=..., profile_batch=(5, 10)). Abrir Profiler tab.
5. Bottleneck: si la GPU está al 30 %, el bottleneck es data loading. Optimizar pipeline (clase 109).

Homework verificable

Optimizar el training de un modelo:

1. Baseline con default config.
2. Activar mixed precision + memory_growth.
3. Profile y identificar bottleneck.
4. Aplicar fix (más prefetch, batch más grande, etc.).
5. Reportar speedup wall-clock.

Criterio de aceptación: speedup ≥ 1.5× con mixed precision; identificado el bottleneck correctamente.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
Could not find cuda drivers	Drivers no instalados o versión incompatib
OOM al usar mixed precision	A veces el ahorro de memoria no es total.
mixed_float16 loss = NaN	Loss scaling automático debe estar activo.
GPU al 100 % CPU al 100 % también	Data loading no es el bottleneck → OK. Si
Múltiples GPU pero usa 1	No configuraste strategy. Fix: MirroredStr

Preguntas frecuentes

¿float16 o bfloat16?

bfloat16 si GPU lo soporta (Ampere+, A100/H100, TPU). Más estable, sin loss scaling. float16 en GPUs viejas (V100, T4).

¿GPU consumer (RTX) para DL profesional?

Sí — RTX 4090 / 5090 tienen 24 GB VRAM y excelente performance. Para LLMs grandes (>30B), considerar A100/H100.

¿Cuántas GPU necesito?

1 GPU para experimentos. Multi-GPU para datasets grandes (ImageNet) o modelos grandes (LLMs).

¿Apple Silicon (M-chips)?

TF tiene soporte vía tensorflow-metal. PyTorch via MPS backend. Para experimentos chicos OK; producción

serio sigue siendo NVIDIA o TPU.

¿GPU en cloud?

AWS p4/p5, GCP A2/A3, Lambda Labs, Vast.ai. Lambda/Vast son más baratos para experimentos.

Referencias

- Géron, cap. 19 — Using GPUs to Speed Up Computations.
- TF GPU guide.
- Mixed precision guide.
- NVIDIA H100 / Hopper architecture whitepaper.

Siguiente clase

Clase 172 — Entrenamiento multi-dispositivo, tf.distribute

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb