Clase 144 — Flash Attention v2/v3, RoPE, GQA: el motor de los LLMs modernos

Parte: 2 — Deep Learning · Fuente: Dao et al. (2022, 2023, 2024) FlashAttention + Su et al. (2021) RoPE + Ainslie et al. (2023) GQA. Duración estimada: 90 min.

Clase 144 — Flash Attention v2/v3, RoPE, GQA: el motor de los LLMs modernos

> *Parte: 2 — Deep Learning · Fuente: Dao et al. (2022, 2023, 2024) FlashAttention + Su et al. (2021) RoPE + Ainslie et al. (2023) GQA. Duración estimada: 90 min.*

Objetivo

Entender en profundidad las 3 piezas técnicas que hacen que un LLM moderno (Llama 3, Mistral, Qwen, Gemma) sea rápido y memory-efficient: Flash Attention v2/v3 (O(N) memoria + 2-3× speedup), Rotary Position Embeddings (RoPE) (mejor extrapolación), Grouped-Query Attention (GQA) (menos KV cache en inference).

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Explicar por qué attention naïve es O(N²) en memoria y cómo FlashAttention lo reduce a O(N) con online softmax + tiling.
- Implementar RoPE: rotar pares de dimensiones de Q, K por ángulo función de posición.
- Diferenciar MHA, MQA, GQA — y por qué GQA es el compromiso default 2026.
- Aplicar torch.nn.functional.scaled_dot_product_attention(q, k, v, is_causal=True) que elige Flash auto.
- Reconocer combinación moderna: RMSNorm + GQA + RoPE + SwiGLU + Flash Attention.

Temas

- Attention cost: matriz (N, N) → 64 MB por head con N=8192, fp16.
- FlashAttention: bloques en SRAM, no materializa la matriz completa.
- v1 (2022), v2 (2023, 2× speedup), v3 (2024, optimizado H100).
- Positional encoding: sinusoidal → learnable → RoPE.
- RoPE: rotación bidimensional, propiedad relativa.
- MHA / MQA / GQA: trade-off entre calidad y memoria.
- KV cache: por qué crece en inference.

Definiciones y características

- FlashAttention: algoritmo IO-aware. Reformula softmax(QK^T)V en bloques que caben en SRAM.
- Online softmax: actualización incremental del softmax sin materializar todo.
- RoPE: q' = R_θ q donde R_θ rota pares de dims. θ_i = 10000^(-2i/d).
- MHA: H_q = H_kv (clásico).
- GQA: H_kv = H_q / G. G grupos. Llama 2 70B usa G=8.
- MQA: H_kv = 1. Extremo de GQA.

Dataset / recursos

- HuggingFace modelos: Llama 3, Mistral 7B.
- Librerías: flash-attn, torch ≥ 2.0 (SDPA), transformers.

Ejercicios

1. SDPA vs naïve: implementar attention naïve y F.scaled_dot_product_attention. Benchmark.
2. RoPE: implementar rotation function, verificar propiedad attention(R_θ q, R_φ k) = f(θ - φ).
3. GQA Vs MHA: con Llama config (n_heads=32, kv_heads=8), inspeccionar shapes.
4. KV cache: medir VRAM en inference con secuencia 8192 — comparar MHA vs GQA.
5. FlashAttention v3 en H100: si tenés H100, benchmark vs v2.

Homework verificable

Mini-GPT con piezas modernas:

1. 6-layer Transformer con: RMSNorm, GQA (4 KV heads / 8 Q heads), RoPE, SwiGLU FFN.
2. Train next-token sobre Tiny Shakespeare.
3. Comparar contra mini-GPT clásico (LayerNorm + MHA + Sin PE + GELU FFN).

Criterio de aceptación: mini-GPT moderno entrena más estable + menor memoria; quality comparable.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
flash-attn no instala en CUDA viejo	Requiere CUDA 11.6+, GPU Ampere+. Fix: usa
RoPE con base distinta a 10000	Para extrapolación a contextos largos (32k
MQA da peor calidad	Esperado. Fix: GQA es el compromiso.
KV cache OOM en context largo	Inherente. Fix: GQA + quantization (Q8 KV)
is_causal=True en SDPA solo aplica si tens	Fix: passing attn_mask cuando shapes asimé

Preguntas frecuentes

FlashAttention v2 o v3?

v3 si tenés H100. v2 estable para todo lo demás. SDPA de PyTorch elige el mejor disponible.

RoPE absoluto o relativo?

RoPE codifica posición absoluta pero produce comportamiento relativo en attention. Lo mejor de ambos.

GQA en training?

Sí — entrenar con GQA desde el principio. Llama 2 70B y todos los modernos lo hacen.

Combina con sliding window attention?

Sí — Mistral 7B usa GQA + sliding window. Para contextos infinitos.

¿Y para CV (ViT)?

ViT moderno también usa Flash Attention (timm support). RoPE en algunos (DiT). GQA menos común en CV.

Referencias

- Dao et al. (2022, 2023, 2024), FlashAttention v1/v2/v3.
- Su et al. (2021), RoFormer: Enhanced Transformer with Rotary Position Embedding.
- Ainslie et al. (2023), GQA: Training Generalized Multi-Query Transformer Models.
- Touvron et al. (2023), Llama 2.

Siguiente clase

Clase 145 — Hugging Face Transformers (uso práctico)

Apéndice: notebook (primer bloque)

Los 3 ingredientes que hacen que un LLM moderno (Llama-3, Mistral) corra en GPU consumer. Todo implementado en numpy puro.

```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(42)

n, d = 32, 64
Q = np.random.randn(n, d) / np.sqrt(d)
K = np.random.randn(n, d) / np.sqrt(d)
V = np.random.randn(n, d)
print('Q,K,V shapes:', Q.shape, K.shape, V.shape)
```

Archivos complementarios

- notebook.ipynb