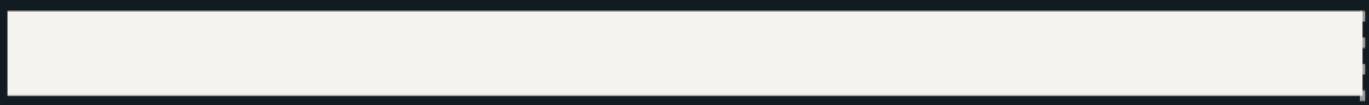
Qué modelo cabe en tu gráfica

ENCARGO INGENUO 85/120



BRIEF OPERATIVO 120/120



Qué modelo de IA cabe en tu tarjeta gráfica

Cuánta VRAM y RAM tienes de verdad, qué modelos y cuantizaciones caben, a qué velocidad esperar y cómo medirlo tú antes de gastar dinero

Edición: 05/11/2026. **Para quién:** tienes un PC con Windows (o vas a comprar uno) y quieres usar modelos de IA en tu propio equipo, con Ollama o LM Studio. Antes de descargar 20 GB o de pagar una gráfica nueva, quieres saber qué cabe y a qué velocidad irá. Sabes abrir PowerShell y pegar un comando; no hace falta que programes ni que sepas cómo funciona un modelo por dentro.

Alcance

Todo lo medido en este libro se midió el 04/11/2026 en un solo equipo: Windows 11 Pro, Intel Core Ultra 5 245K, 31 GB de RAM y una NVIDIA RTX 5060 Ti de 16 GB, con Ollama 0.40.14 y los modelos qwen3.5 (4B y 9B), gemma4 (12B) y qwen3.6 (35B). Las velocidades son de ese equipo. Lo que no medimos (otras memorias, otras gráficas, AMD, integradas, Mac) sale de la regla de cálculo y de la documentación oficial, y en cada caso lo marcamos como estimación. Es el único aviso de este tipo que vas a leer.

Cómo usar este libro

Empieza por el capítulo 01: en menos de una hora tendrás tu VRAM libre real y un modelo medido en tu equipo. Si después solo quieres la respuesta, salta a la tabla del capítulo 07. Si quieres entender la cuenta para cualquier modelo, los capítulos 03 a 05 la dan pieza a pieza, con la hoja `estimar-memoria.csv` del paquete.

Lo que quieres saber	Capítulo
Cuánta memoria de gráfica tienes y cuánta te queda libre	01
Si cuenta la memoria compartida, la integrada o la de un Mac	02
Cuánto ocupa un modelo antes de descargarlo	03
Qué cuantización bajar (Q4, Q5, Q8)	04
Cuánto cuesta subir el contexto	05
Qué velocidad esperar y cómo medirla bien	06
Qué modelo probar primero con tu equipo, y qué mirar antes de comprar	07

Este libro decide qué cabe. Si un modelo que debería caber da error, no usa la gráfica o responde vacío, eso se diagnostica en Ollama en Windows cuando falla, al que remitimos cuando toca.

Qué necesitas

- Windows 10 u 11 con PowerShell (viene con Windows).
- La carpeta `scripts` del paquete en tu equipo, por ejemplo en `C:\ia-grafica`. Windows bloquea los scripts descargados: desbloquéalos una vez con `Get-ChildItem C:\ia-grafica | Unblock-File` y lánzalos con `powershell -ExecutionPolicy Bypass -File .\inventario-ia.ps1`.
- Para medir velocidades: Ollama instalado y unos 3 GB libres para el primer modelo.
- Para la hoja: Excel o LibreOffice Calc.

Contenido

Qué modelo de IA cabe en tu tarjeta gráfica . . . 2

Cuánta VRAM y RAM tienes de verdad, qué modelos y cuantizaciones caben, a qué velocidad esperar y cómo medirlo tú antes de gastar dinero . . . 2

Alcance . . . 2

Cómo usar este libro . . . 2

Qué necesitas . . . 2

01 · En 60 minutos: tu VRAM de verdad y el primer modelo medido en tu equipo 5

Paso 1 (10 minutos): el inventario . . . 5

Paso 2 (10 minutos): lo que ya está ocupado . . . 6

Paso 3 (15 minutos): la cuenta que decide si un modelo cabe . . . 6

Paso 4 (20 minutos): mide un modelo en tu equipo . . . 7

Paso 5 (5 minutos): apúntalo . . . 7

Cómo sabes que ha salido bien . . . 7

Resumen . . . 8

02 · Qué memoria cuenta: VRAM, RAM, compartida, integradas y Mac 9

Dedicada y compartida, según Microsoft . . . 9

Lo que pasa cuando el modelo no cabe . . . 9

La RAM también cuenta, aunque todo quepa . . . 9

Gráficas integradas . . . 10

Portátiles . . . 10

NVIDIA y AMD en Windows . . . 10

Mac con Apple Silicon . . . 11

LM Studio en Windows . . . 11

Cómo compruebas que lo has entendido en tu equipo . . . 11

Resumen . . . 11

03 · Cuánto ocupa un modelo: la cuenta de los pesos 12

La cuenta . . . 12

Comprobado con el registro . . . 12

Dónde quedó cada parte . . . 13

Léelo en tu registro . . . 13

Antes de descargar: la cuenta en la hoja . . . 14

Los modelos «MoE» . . . 14

Cómo compruebas la cuenta en tu equipo . . . 14

Resumen . . . 15

04 · Cuantizaciones: Q4, Q5, Q8 y qué pierdes 16

Qué es, en una frase . . . 16

La tabla de llama.cpp . . . 16

Cuánto empeora, según llama.cpp . . . 17
Lo que vimos en el laboratorio . . . 17
Qué bajar, según tu memoria . . . 17
Cómo reconoces la cuantización . . . 18
Comprueba la diferencia en tu trabajo . . . 18
Resumen . . . 18

05 · El contexto también ocupa: la caché KV **19**

Qué es la caché KV . . . 19
La fórmula . . . 19
Lo que midió el registro . . . 19
Qué contexto pone Ollama por defecto . . . 20
Comprimir la caché: q8_0 y q4_0 . . . 21
Cuánto contexto necesitas tú . . . 21
Cómo compruebas el coste en tu equipo . . . 21
Resumen . . . 21

06 · Velocidad: qué esperar y cómo medirla **22**

Dos velocidades, no una . . . 22
Lo que medimos . . . 22
La carga del modelo . . . 23
Cuando no cabe: el 35B . . . 23
Cuando la gráfica se llena: de 111 a 3,8 . . . 23
Mide sin engañarte . . . 23
Otras gráficas . . . 24
Cómo compruebas que tu medición vale . . . 24
Resumen . . . 24

07 · Tu equipo → qué probar primero **25**

Las cifras de partida . . . 25
La tabla . . . 25
Para un modelo que no está en la tabla . . . 26
Antes de comprar . . . 26
Resumen del libro . . . 27
Fuentes . . . 27

01 · En 60 minutos: tu VRAM de verdad y el primer modelo medido en tu equipo

Al acabar este capítulo sabrás cuánta memoria de gráfica tienes de verdad (no la que dice la caja ni la que dice cierta ficha de Windows), cuánta te queda libre con tus programas abiertos, qué modelo cabe con holgura y a qué velocidad escribe en tu equipo. Lo vas a comprobar tú, no a creértelo.

Necesitas Windows 10 u 11, conexión a internet para una descarga de unos 3 GB y la carpeta `scripts` del paquete copiada en tu equipo, por ejemplo en `C:\ia-grafica`.

Paso 1 (10 minutos): el inventario

Windows bloquea los scripts descargados. Desbloquéalos una vez y lanza el inventario:

```
Get-ChildItem C:\ia-grafica | Unblock-File
cd C:\ia-grafica
powershell -ExecutionPolicy Bypass -File .\inventario-ia.ps1
```

El script solo lee: no instala nada, no cambia nada y no se conecta a internet. Te muestra el procesador (y si tiene AVX2), la RAM total y libre, cada gráfica con su memoria, la memoria de gráfica que ya está ocupada, el espacio libre de tus discos y si ya tienes Ollama o LM Studio. Si añades `-Guardar`, deja también un `inventario-ia.json` para comparar más adelante.

Esto es lo que dio en el equipo del laboratorio de este libro:

```
== Graficas ==
Intel(R) Graphics - INTEGRADA - memoria reservada: 2 GB (usa la RAM del sistema)
NVIDIA GeForce RTX 5060 Ti - DEDICADA - VRAM: 15,9 GB - controlador 32.0.16.2101
nvidia-smi: NVIDIA GeForce RTX 5060 Ti - 15,9 GB en total, 3,3 GB ocupados ahora - controlador 621.01
```

Dos gráficas: la integrada del procesador y la dedicada. Para la IA local cuenta la dedicada. La integrada no tiene memoria propia: lo que aparece es RAM del sistema reservada para ella (capítulo 02).

La trampa de los 4 GB

Hay una forma muy extendida de preguntar a Windows por la gráfica, la clase `Win32_VideoController`. Pruébala:

```
Get-CimInstance Win32_VideoController | Select-Object Name, AdapterRAM
```

En el equipo del laboratorio, para una tarjeta de 16 GB, respondió `4293918720`: 4 GB. No es una avería. La documentación de Microsoft declara ese campo como un número de 32 bits sin signo (`uint32`), y el mayor número que cabe en 32 bits ronda los 4.000 millones de bytes. Ninguna gráfica de más de 4 GB puede aparecer ahí con su memoria real. Por eso el script lee la cifra del registro de Windows (`HardwareInformation.qwMemorySize`) y, si la gráfica es NVIDIA, la contrasta con `nvidia-smi`.

Si alguna vez una herramienta te dice que tu gráfica «tiene 4 GB» y sabes que tiene más, ya sabes de dónde sale.

Si no quieres usar el script

- **Administrador de tareas** (Ctrl + Mayús + Esc) → Rendimiento → GPU. «Memoria de GPU dedicada» es la VRAM: el total y lo que está en uso.
- **NVIDIA**: `nvidia-smi --query-gpu=name,memory.total,memory.used --format=csv` viene con el controlador.

Paso 2 (10 minutos): lo que ya está ocupado

Tu modelo no dispone de toda la VRAM. El escritorio, el navegador y cualquier programa que use la gráfica ya tienen su parte. En el equipo del laboratorio, con el escritorio y aplicaciones normales abiertas, había entre 1,5 y 3 GB ocupados antes de cargar ningún modelo.

Abre lo que uses mientras trabajas (navegador con tus pestañas, editor, lo que sea) y vuelve a lanzar el inventario. La línea final te da la resta:

```
VRAM para modelos (total - ocupada ahora): 12,6 GB.
```

Ese es tu número de trabajo. Apúntalo. Si juegas o editas vídeo en el mismo equipo, hazlo también con eso abierto: verás que el número baja mucho, y en el capítulo 06 verás qué pasa cuando no queda sitio.

Paso 3 (15 minutos): la cuenta que decide si un modelo cabe

Un modelo cargado ocupa tres cosas en la gráfica:

1. **Los pesos**: el modelo en sí. Se calcula con parámetros × bits por peso (capítulo 03).
2. **El contexto**: la memoria donde guarda el texto que está leyendo y escribiendo, la «caché KV». Crece con el contexto que configures (capítulo 05).
3. **Un margen de cálculo**: unos cientos de MB para las operaciones.

La regla de bolsillo, antes de afinar:

> pesos + contexto + margen ≤ VRAM libre (la del paso 2)

Si se cumple con holgura, el modelo irá entero en la gráfica. Si no, Ollama reparte capas entre gráfica y RAM: funciona, pero más despacio, y a veces no funciona (capítulo 06). Abre `estimar-memoria.csv` con Excel o LibreOffice y mira las primeras filas: son los tres modelos del laboratorio, con la estimación al lado de lo que midió el registro de Ollama.

Estos son los modelos que probamos el 04/11/2026 en el equipo del laboratorio (Windows 11, Intel Core Ultra 5 245K, 31 GB de RAM, RTX 5060 Ti de 16 GB, Ollama 0.40.14), con su velocidad de escritura con un prompt de 524 tokens:

Modelo	Escribe con la gráfica	Escribe solo con CPU
qwen3.5:4b	92–95 t/s	11,9 t/s
qwen3.5:9b	56–58 t/s	5,7 t/s
gemma4:12b	111–113 t/s	sin medir
qwen3.6:35b	no fue utilizable	sin medir

«t/s» son tokens por segundo; un token es un trozo de palabra. Fíjate en dos cosas que nadie te cuenta en una tabla de requisitos:

- **El más grande de los tres que cupieron fue el más rápido.** gemma4:12b escribió casi el doble de rápido que qwen3.5:9b. El tamaño no predice la velocidad; por eso se mide.
- **El de 35B no entró.** Sus pesos pasan de 20 GiB, y en 16 GB de VRAM Ollama lo repartió entre gráfica y RAM. Respondió una vez y después falló con error 500. Lo que ocupa cada modelo, parte por parte, está en el capítulo 03.

Paso 4 (20 minutos): mide un modelo en tu equipo

Si no tienes Ollama, descárgalo de ollama.com e instálalo con el instalador para Windows. Según su documentación, necesita Windows 10 22H2 o posterior y, con NVIDIA, el controlador 551.61 o más reciente. Después, en PowerShell:

```
ollama pull qwen3.5:4b
cd C:\ia-grafica
powershell -ExecutionPolicy Bypass -File .\medir-velocidad.ps1 -Modelo qwen3.5:4b
```

El primer modelo es el pequeño a propósito: cabe en casi cualquier gráfica dedicada actual y te deja ver si todo funciona. `medir-velocidad.ps1` hace tres peticiones, cambia el principio del texto en cada una para que Ollama no reutilice lo ya leído y termina así:

```
qwen3.5:4b - contexto 8192 - con grafica si cabe
Escribe: ... t/s de media (...)   Lee: ... t/s de media
Donde esta cargado (ollama ps): 100% GPU
VRAM ocupada: ... MiB antes -> ... MiB con el modelo cargado
```

Lo que tienes que mirar:

- **«100% GPU»**: el modelo está entero en la gráfica. Si pone algo como `30%/70% CPU/GPU`, parte está en la RAM.
- **«Escribe»**: tu velocidad real. Compárala con la tabla del paso 3. Si estás en el orden de la columna «solo con CPU» teniendo gráfica dedicada, Ollama no la está usando: es un fallo, no un límite de tu equipo, y se diagnostica con el libro Ollama en Windows cuando falla, capítulo 02.
- **La primera prueba tarda más**: incluye cargar el modelo del disco a la memoria. Las siguientes, no.

Ahora repite con la gráfica desactivada para ver la diferencia en tu equipo:

```
powershell -ExecutionPolicy Bypass -File .\medir-velocidad.ps1 -Modelo qwen3.5:4b -SoloCPU
```

Con esas dos cifras ya sabes cuánto te da la gráfica a ti.

Paso 5 (5 minutos): apúntalo

Guarda las dos mediciones en un archivo que irás llenando durante el libro:

```
.\medir-velocidad.ps1 -Modelo qwen3.5:4b -Anotar .\mis-mediciones.csv
.\medir-velocidad.ps1 -Modelo qwen3.5:4b -SoloCPU -Anotar .\mis-mediciones.csv
```

Y apunta a mano tu VRAM libre del paso 2. Con esos tres datos, el capítulo 07 te dice qué modelo probar después y cuál no merece la descarga.

Cómo sabes que ha salido bien

- Tienes tu VRAM total y tu VRAM libre con tus programas abiertos, y sabes que el «4 GB» de `AdapterRAM` no es tu cifra.
- `medir-velocidad.ps1` dice `100% GPU` con qwen3.5:4b y una velocidad de escritura claramente por encima de la de `-SoloCPU`.
- Tienes `mis-mediciones.csv` con dos filas.

Si el modelo no carga, da error 500 o la respuesta llega vacía, no sigas probando modelos más grandes: el fallo no es de tamaño. Ese terreno lo cubre Ollama en Windows cuando falla.

Resumen

Tu VRAM útil es la total menos lo que ya ocupan tus programas (en el laboratorio, entre 1,5 y 3 GB). `Win32_VideoController` nunca informa de más de 4 GB porque su campo es de 32 bits. Un modelo cabe si pesos + contexto + margen caben en esa VRAM libre. Y la velocidad no se deduce del tamaño: gemma4:12b, el mayor de los que cupieron, fue el más rápido. Mídelo en tu equipo con `medir-velocidad.ps1`.