The 8GB VRAM Trap: What 'Runs on 8GB' Actually Means

February 25, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** Yes, 7B-8B models run on 8GB VRAM at Q4_K_M quantization — 40-70 tokens/sec depending on the GPU, genuinely useful for chat and simple tasks. But the fine print matters. Your context window tops out around 16K tokens (not the 128K the model claims). You can't run an embedding model alongside your chat model for RAG. Vision models mostly won't fit. SDXL barely works. Fine-tuning is theoretically possible but practically too tight. And that 70B model someone told you 'runs on 8GB' at Q2 quantization? Its perplexity score jumps from 3.1 to 11.9 and it loses 30 percentage points on reasoning benchmarks. It loads. It does not work. The real entry point for serious local AI is 12GB — a used RTX 3060 12GB at $275 costs $25 more than a used RTX 4060 8GB and gives you 50% more VRAM.

📚 **Related:** What Can You Run on 8GB VRAM · VRAM Requirements Guide · Quantization Explained · What Can You Run on 12GB · GPU Buying Guide · Planning Tool

"Runs on 8GB VRAM" is the "fits in a carry-on" of local AI. Technically true. Practically, you're stuffing a week's worth of clothes into a bag designed for a weekend, and the zipper is about to blow.

Every beginner guide, every Reddit comment, every YouTube thumbnail promises you can run local LLMs on an 8GB GPU. And you can. A 7B model at Q4 quantization loads, generates text, and gives you real results at 40-70 tokens per second. That part is honest.

What nobody mentions: your context window is a fraction of what the model advertises, you can't run two models at once, vision models mostly won't fit, and that 70B model someone quantized to Q2 so it would "run on 8GB" produces output that fails basic reasoning tests. This article is the fine print.

The marketing claim vs the reality

Here's the thing that makes 8GB VRAM misleading: the model file fits. An 8B model at Q4_K_M is about 4.7GB. That's well under 8GB. So tutorials say "runs on 8GB" and move on.

But the model file isn't the only thing in VRAM. You also need:

Component	VRAM	Notes
Model weights (8B Q4_K_M)	~4.7 GB	The .gguf file
CUDA overhead	~0.5 GB	Driver, compute buffers
Scratchpad / compute buffer	~0.6 GB	Intermediate calculations
KV cache (context)	Varies	Grows with every token

That leaves about 2.2GB for your context window. At roughly 0.1GB per 1K tokens of context for an 8B model, that's about 16K tokens of usable context before you hit the wall.

The model claims 128K context. You get 16K. That's not a rounding error — it's an 8x gap between the spec sheet and reality.

The quantization quality ladder

Quantization compresses model weights to use less VRAM. The tradeoff is quality loss. Here's where the cliff is, measured on Llama 3.1 8B:

Quant	Model Size	Perplexity	MMLU Score	Quality
FP16 (baseline)	15.3 GB	7.32	63.5%	Full quality
Q8_0	8.1 GB	7.33	63.4%	Indistinguishable
Q6_K	6.3 GB	7.35	63.2%	Negligible loss
Q5_K_M	5.5 GB	7.40	62.8%	Minor loss
Q4_K_M	4.7 GB	7.56	62.4%	The sweet spot for 8GB
Q3_K_M	3.8 GB	7.96	62.0%	Noticeable degradation
Q3_K_S	3.5 GB	8.96	59.3%	The cliff starts here
Q2_K	~3.0 GB	~210	~36%	Garbage

Source: "Which Quantization Should I Use?" (arXiv:2601.14277) and "How Good Are Low-bit Quantized LLaMA3 Models?" (arXiv:2404.14047)

Q4_K_M loses about 1 MMLU point from full precision. That's the sweet spot — 69% size reduction with minimal quality loss. Q3_K_S is where things start falling apart: 4 points of MMLU lost, perplexity jumping 22%.

Q2 is a cliff, not a slope. Perplexity goes from 7.56 at Q4 to 210 at Q2 — the model produces statistically random text. Community blind tests on llama.cpp (500+ votes) found that below IQ3_XS, quality loss becomes obvious to anyone reading the output.

When someone says a 70B model "runs on 8GB" at Q2, this is what they mean: it loads into memory and generates tokens. Those tokens are roughly as useful as a random word generator.

What 8GB VRAM actually gets you

Here's the honest list, tested on RTX 4060 (8GB) and RTX 3060 Ti (8GB):

Model	Quant	VRAM Used	Speed	Verdict
Llama 3.1 8B	Q4_K_M	~5.5 GB	42-57 tok/s	Good for chat, summarization
Qwen 2.5 7B	Q4_K_M	~5.3 GB	42-58 tok/s	Strong multilingual, coding
Mistral 7B	Q4_K_M	~4.7 GB	51-71 tok/s	Fast, good general use
Phi-4 Mini (3.8B)	Q4_K_M	~2.8 GB	80+ tok/s	Best reasoning per VRAM
Gemma 2 9B	Q4_K_M	~6.0 GB	18-24 tok/s	Tight fit, slow

Sources: DatabaseMart GPU benchmarks (RTX 4060, RTX 3060 Ti)

These are genuinely useful models. Qwen 2.5 7B writes decent code. Mistral 7B handles summarization well. Phi-4 Mini punches above its weight on reasoning. For chat, simple coding, and summarization, 8GB works.

The constraints:

You get ~16K context max. Hardware Corner benchmarks show an 8B model at Q4 consuming 5GB at 4K context, 6GB at 8K, 7GB at 16K, and 9GB at 32K. That 32K figure overflows your 8GB and triggers CPU offloading, which tanks speed from 50 tok/s to 5-10 tok/s.

One model at a time. Loading an embedding model alongside a chat model for RAG requires squeezing both into 8GB. A tiny embedding model like nomic-embed-text (~262MB) can technically coexist with a 7B chat model, but only at 2-4K context. Any real RAG pipeline needs more headroom.

Your GPU is also running your display. If the 8GB card drives your monitor, subtract ~0.3-0.5GB for the desktop compositor. Generation speed drops when your GPU is multitasking.

Slow generation on anything that barely fits. Gemma 2 9B loads on 8GB but runs at 18-24 tok/s — under half the speed of Mistral 7B. The model fits, but there's no room left for compute buffers to work efficiently.

What 8GB VRAM does not get you

70B+ models at usable quality

The math: Llama 3.3 70B at Q4_K_M is 42.5GB. Even the most aggressive quantization (IQ2_XXS) is 19.1GB — more than double your VRAM. You'd need to CPU-offload 60-75% of layers, dropping speed to 1-3 tokens per second. And the quality loss at Q2 is catastrophic: the 70B model loses 30 percentage points on CommonSenseQA (76% → 45%) and perplexity quadruples.

Loading a 70B model on 8GB VRAM is not "running it slowly." It's running a broken version of the model slowly.

Long context

Your KV cache grows with every token in the conversation. For an 8B model:

Context Length	Total VRAM Needed	Fits on 8GB?
4K tokens	~5 GB	Yes
8K tokens	~6 GB	Yes
16K tokens	~7 GB	Barely
32K tokens	~9 GB	No
128K tokens	~23 GB	Not a chance

The model's spec sheet says "128K context." Your GPU says 16K. Believe the GPU.

Vision models

Vision models need VRAM for the image encoder on top of the language model. Even the small ones are tight:

Model	VRAM Needed	Works on 8GB?
LLaVA 7B (Q4)	~5-6 GB	Yes, but limited context
Qwen2.5-VL 3B (FP16)	~6 GB	OOMs without optimization
Qwen2.5-VL 7B	8+ GB	Not without heavy quantization

LLaVA 7B at Q4 is the only vision model that works reliably on 8GB, at 72 tok/s on a 3060 Ti. Qwen2.5-VL 3B — a model with only 3 billion parameters — OOMs at FP16 on 8GB GPUs without reducing image resolution and applying torch.compile() optimizations.

Multiple models simultaneously

RAG needs an embedding model plus a chat model. Agent workflows need a planning model plus a tool-calling model. Multi-model setups need VRAM for… multiple models. On 8GB, you get one model with room to breathe, or two models gasping for air.

Fine-tuning

Method	VRAM for 7B model
Full fine-tuning	~70 GB
LoRA	~15 GB
QLoRA (8-bit)	~9 GB
QLoRA (4-bit)	~5 GB

4-bit QLoRA on a 7B model technically fits in 8GB at ~5GB. But you have almost no headroom for batch size, context length, or gradient accumulation. The result is slower training, smaller batches, and measurably worse outcomes — QLoRA-trained Llama 3 8B scored 56.7% on MMLU versus 64.8% for the FP16 fine-tune. 12-16GB is the realistic minimum for QLoRA that produces good results.

Image generation beyond SD 1.5

Model	Minimum VRAM	On 8GB?
Stable Diffusion 1.5	4-6 GB	Yes, comfortably
SDXL (base only)	8 GB	Barely. ~30 sec/image
SDXL (base + refiner)	12-16 GB	No

Model	Minimum VRAM	On 8GB?
Flux (full)	12+ GB	No
Flux NF4 (quantized)	6-8 GB	Yes, with CPU workarounds

SD 1.5 runs great on 8GB. SDXL technically fits but is painfully slow and can't use the refiner. Full Flux needs 12GB+. Quantized Flux (NF4) works with –lowvram flags and CPU text encoder offloading, but generation times are measured in minutes, not seconds.

The real minimum for common tasks

Task	Minimum for acceptable quality	Why
Chat / assistant	8GB (7B at Q4)	Works well for simple conversations
Coding assistant	12GB+ recommended	Smarter models (14B+) make fewer mistakes
RAG / document Q&A	12GB+ (embedding + chat)	Two models need to coexist
Image generation (SD 1.5)	8GB	Comfortable fit
Image generation (SDXL/ Flux)	12GB+	Base SDXL barely fits, Flux needs more
Voice / TTS	8GB usually fine	Whisper + Kokoro TTS are small
Translation	8GB (NLLB 1.3B)	Dedicated translation models are tiny
Vision / image understanding	12GB+	Even small VL models are tight on 8GB
Fine-tuning (QLoRA)	12GB+ practical	8GB is too tight for decent results

8GB works for three things: basic chat, SD 1.5 image generation, and small specialized models (TTS, translation). Everything else wants 12GB or more.

The upgrade path

If you're on 8GB and hitting walls, here's what to actually buy:

A used RTX 3060 12GB runs $275 on eBay. It's slower than a 4060 at inference, but the extra 4GB of VRAM lets you run 7B models at 32K context, load an embedding model alongside a chat model, and run SDXL without –lowvram hacks. For $25 more than a used 4060, you get 50% more VRAM. That's the real entry point for local AI beyond toy projects.

A used RTX 3090 24GB runs $900 on eBay. This is where local AI gets serious: 32B models at Q4, 70B models at Q3, LoRA fine-tuning with real batch sizes, Flux at full resolution. If you're going to spend money, this is the sweet spot. See the used RTX 3090 buying guide for what to watch for.

A Mac with Apple Silicon uses unified memory, so your system RAM doubles as VRAM. An M4 Mac Mini with 32GB runs 7B models at 30+ tok/s and 32B models at reasonable speed. Slower than NVIDIA for inference, but the VRAM ceiling is much higher. See the Mac Mini M4 guide for specifics.

Use the Planning Tool to see what fits on your hardware before you buy.

How to maximize what you have

If you're stuck on 8GB for now, here's how to get the most out of it.

Use Q4_K_M quantization. It's the best tradeoff: 69% size reduction from FP16 with only 1 MMLU point of quality loss. Don't go below Q3_K_M unless you want to watch your model forget how to reason.

Reduce context length explicitly. In Ollama, set `num_ctx` to 4096 or 8192 instead of letting it auto-detect. Shorter context means more VRAM for compute buffers, which means faster generation:

```
ollama run llama3.1:8b-instruct-q4_K_M --num-ctx 4096
```

Use llama.cpp with partial GPU offloading. If a model is slightly too large for full GPU inference, offload most layers to the GPU and a few to CPU. Partial offload at 80% GPU is much faster than full CPU inference:

```
llama-cli -m model.gguf -ngl 28 -c 4096
```

The `-ngl` flag controls how many layers go to the GPU. For a 32-layer 8B model on 8GB VRAM, try 28 layers on GPU, 4 on CPU. Adjust up or down until you stop hitting OOM.

Close other GPU-using apps. Chrome with hardware acceleration eats 0.5-1GB of VRAM. VS Code's GPU renderer takes another chunk. Close them or disable hardware acceleration before running inference.

Consider CPU inference for occasional use. If you have 32GB+ of system RAM, you can run larger models on CPU at 3-8 tok/s. Slow, but it works for one-off tasks where you need a smarter model than your GPU can hold. See the CPU inference guide for what to expect.

Pick the right model for the job. Phi-4 Mini (3.8B) fits easily on 8GB and outperforms models twice its size on reasoning. Mistral 7B is the fastest 7B option. Qwen 2.5 7B is the best all-rounder. Don't default to the biggest model that fits — default to the one that's best at what you need.

The bottom line

8GB VRAM is not useless for local AI. A 7B model at Q4_K_M runs at 40-70 tokens per second, handles basic chat and summarization, and costs nothing per query. For trying out local AI and deciding if you want to go further, it's a legitimate starting point.

But 8GB is a starting point, not a destination. The moment you need longer context, smarter models, RAG, vision, or image generation beyond SD 1.5, you're hitting walls. The guides that say "runs on 8GB" aren't lying — they're just leaving out the asterisk.

The smallest useful upgrade is a used RTX 3060 12GB for $275. The real sweet spot is a used RTX 3090 at 24GB. Check the GPU buying guide and the used GPU guide for current prices and what to look for.
