Qwen 3.5 Locally — 27B vs 35B-A3B vs 122B, Which Model Fits Your GPU

February 26, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** The 35B-A3B MoE is the standout. It generates at 111 tok/s on an RTX 3090 — 3x faster than the 27B dense — because only 3B of its 35B parameters activate per token. Needs ~22GB at Q4, so a 24GB GPU handles it. The 27B dense uses less memory (~17GB at Q4) and is the safer pick for tighter cards. The 122B-A10B needs ~70GB — Mac Studio or multi-GPU territory. All three are multimodal with 262K context and Apache 2.0. Get the 35B MoE if you have 24GB VRAM. Get the 27B if you have less.

📚 **More on this topic:** Qwen 3.5 397B Guide · Qwen 3.5 9B Setup Guide · Qwen 3.5 Mac: MLX vs Ollama · Qwen Models Guide · VRAM Requirements · Running LLMs on Mac

Qwen 3.5 shipped four model sizes. The 397B flagship gets the headlines, but it needs 192GB+ of memory. Most people don't have that.

The three models that run on consumer hardware: **27B dense**, **35B-A3B MoE**, and **122B-A10B MoE**. Same architecture — hybrid attention, 262K native context, built-in vision, Apache 2.0. The difference is how much memory they need and how fast they generate tokens.

The three models at a glance

	27B Dense	35B-A3B MoE	122B-A10B MoE
Total parameters	~28B	~36B	~125B
Active per token	28B (all)	3B	10B
Expert count	None (dense)	256 (8 routed + 1 shared)	MoE
Context window	262K	262K	262K
Vision	Yes	Yes	Yes
License	Apache 2.0	Apache 2.0	Apache 2.0
VRAM at Q4 (short context)	~17 GB	~22 GB	~70 GB

	27B Dense	35B-A3B MoE	122B-A10B MoE
Generation speed (RTX 3090 Q4)	~34 tok/s	~111 tok/s	Multi-GPU only

The speed gap between the 27B and 35B MoE isn't a typo. MoE models only compute through their active parameters per token. The 35B-A3B activates 3B parameters each forward pass — less than the 27B's full 28B — so it generates tokens faster despite having more total weight loaded.

The trade-off: MoE models use more memory for their quality level. You're loading 36B of weights to get 3B of compute. The 27B loads 28B and uses all of it.

VRAM requirements

By model and quantization (at 4K context)

Model	Q8_0	Q6_K	Q4_K_M	Q3_K_M	Q2_K
27B	~30 GB	~23 GB	~17 GB	~14 GB	~11 GB
35B-A3B	~38 GB	~30 GB	~22 GB	~18 GB	~14 GB
122B-A10B	~130 GB	~100 GB	~70 GB	~57 GB	~45 GB

How context length scales VRAM (Q4_K_M)

Model	4K ctx	32K ctx	128K ctx	262K ctx
27B	~17 GB	~19 GB	~27 GB	~33 GB
35B-A3B	~22 GB	~23 GB	~24 GB	~25 GB
122B-A10B	~70 GB	~75 GB	~85 GB	~95 GB

Notice how the 35B-A3B barely grows with context. Going from 4K to 262K only adds ~3GB. That's the hybrid attention system at work — 75% of the layers use Gated DeltaNet (a linear attention variant) which doesn't store traditional KV pairs. The 27B dense also uses hybrid attention, but its larger active parameter count means a bigger KV cache per layer.

On a 24GB GPU, the 35B-A3B at Q4 fits for conversations up to roughly 200K tokens. Beyond that, you'll start hitting the VRAM ceiling. Drop to 128K context if you're seeing OOM errors — still plenty for most workflows.

> Check what fits your setup: VRAM calculator

Speed benchmarks

RTX 3090 (24GB)

Model	Quant	Prompt (tok/s)	Generation (tok/s)
35B-A3B	Q4_K_M	~35	**111**
27B	Q4_K_M	~25	**34**

RTX 5090 (32GB)

Model	Quant	Generation (tok/s)
35B-A3B	Q4_K_M	**165**
27B	Q4_K_M	~50

Apple Silicon (estimated from memory bandwidth)

Model	Quant	Generation (tok/s)	Hardware
35B-A3B	Q4_K_M	~40-50	M4 Max 128GB (MLX)
27B	Q4_K_M	~25-30	M4 Max 128GB (MLX)
122B-A10B	Q4_K_M	~10-15	M4 Max 128GB (MLX)

The 35B MoE at 111 tok/s on an RTX 3090 is faster than most 7B models on the same card. MoE with a 3B active count makes that possible. Mac users can squeeze even more speed out of the 35B with the MLX backend — see our MLX vs Ollama Qwen 3.5 benchmarks for the full comparison.

One caveat: early llama.cpp builds show the 35B-A3B running ~35% slower than its predecessor, the Qwen3-30B-A3B, on CUDA. This appears to be an implementation issue with the new Gated DeltaNet layers, not a fundamental regression. Expect this to improve as llama.cpp adds optimizations. On the MLX backend (Mac), this issue doesn't apply.

35B-A3B: the model most people should run

Three billion active parameters per token means generation runs at roughly 3B speeds. But the model draws from 36B total parameters across 256 experts, giving it a much larger knowledge base. Each token routes to 8 of the 256 experts plus 1 shared expert. Different tokens hit different combinations, so the model uses all its capacity across a conversation — just not all at once per token.

Benchmark scores (thinking mode on)

Benchmark	Score	What it tests
MMLU-Pro	85.3	Broad knowledge
GPQA Diamond	84.2	Graduate-level science
SWE-bench Verified	69.2	Real-world software engineering
AIME 2025	78.0	Math competition
LiveCodeBench v6	78.1	Coding

MMLU-Pro 85.3 and GPQA Diamond 84.2 are scores you'd expect from a much larger dense model. Getting them at 111 tok/s on a consumer GPU is hard to ignore.

Who should pick it

Anyone with a 24GB GPU. The Q4 quantization fits with room for long conversations. If you have an RTX 3090, 4090, or 5090, this is the Qwen 3.5 model to run.

On Mac, it works well on any machine with 48GB+ unified memory. The speed benefit of MoE applies on Apple Silicon too, and the extra memory means you can push context higher.

Where MoE doesn't help

MoE models load more weight per quality level than dense models. The 35B-A3B at Q4 uses ~22GB to get what is effectively 3B of compute per token. A dense 27B at Q4 uses 17GB and all 28B of parameters work on every token.

If your tasks are simple — summarization, translation, short Q&A — the 27B may give comparable quality at lower memory cost. MoE pulls ahead when you need the depth of hundreds of specialized experts: complex reasoning, diverse coding, multilingual work.

27B dense: when less memory is what you have

The 27B is the straightforward option. All parameters active, all the time. No expert routing, no inactive weights sitting in VRAM.

Pick it when:

- **You have 16GB VRAM.** The 35B MoE doesn't fit at Q4. The 27B fits at Q3 (~14GB) with room for moderate context.
- **You want maximum quality per byte loaded.** Every gigabyte of the 27B works on every token. No inactive experts.
- **You run long contexts regularly.** Starting from a lower memory base (17GB vs 22GB) gives you more room for KV cache growth.
- **Predictable latency matters.** Dense models have consistent per-token timing. MoE can vary slightly depending on which experts activate.

The 27B won't match the 35B-A3B's benchmark scores. More total parameters — even when most are inactive per token — does translate to broader capability. But for everyday local inference on constrained hardware, the 27B does the job.

122B-A10B: when you want more quality

The 122B-A10B sits between consumer models and the 397B flagship. Ten billion active parameters per token, 125B total, ~70GB at Q4.

It runs on:

- **Mac Studio M4 Max 128GB** — fits at Q4 with ~58GB free for the OS and apps
- **Mac Studio M3 Ultra 256GB** — fits at Q8 for higher quality
- **2x RTX 3090 (48GB combined)** — fits at Q3 with tensor parallelism
- **1x H100 80GB** — fits at Q4 in VRAM

Ten billion active parameters (vs 3B for the 35B model) means more compute per token and higher quality on hard tasks — complex coding, detailed document analysis, long-form reasoning. If you have the memory and you're pushing into quality-sensitive work, the 122B is the upgrade from the 35B.

Most people don't need it. The 35B-A3B covers the majority of local inference use cases at a fraction of the memory cost.

What's new in Qwen 3.5 (shared across all sizes)

Hybrid attention

Three quarters of the layers use Gated DeltaNet (a linear attention mechanism) and one quarter use standard full attention. This is why the KV cache barely grows with context length and decoding is 8-19x faster than Qwen 3.

262K native context

Base training covers 262K tokens. YaRN scaling extends it to 1M, though quality degrades beyond the native window.

Built-in vision

All Qwen 3.5 models are multimodal from training, not language models with a vision encoder bolted on afterward. For local inference with GGUF files, you still need the mmproj file alongside the model — same two-file setup as Qwen2.5-VL.

Thinking mode on by default

Every Qwen 3.5 model generates chain-of-thought reasoning before answering. Better quality on hard tasks, but costs extra tokens and time. More on how to toggle it below.

Thinking mode: on or off

Qwen 3.5 generates internal reasoning tokens before every response. On by default. Costs extra context and time, but improves accuracy on complex problems.

Keep it on for:

- Math, logic, coding
- Multi-step reasoning
- Tasks where correctness matters more than speed

Turn it off for:

- Simple Q&A, translation, summarization

- Chat where latency matters
- Tasks where extra reasoning doesn't change the answer

How to toggle

Ollama:

```
/set parameter enable_thinking false
```

llama.cpp:

```
llama-cli -m model.gguf \
  --chat-template-kwargs '{"enable_thinking": false}'
```

LM Studio: Look for a thinking mode toggle in model settings. If it's not exposed, set `enable_thinking` to `false` in the inference parameters.

API calls (OpenAI-compatible):

```
{
  "messages": [...],
  "chat_template_kwargs": {"enable_thinking": false}
}
```

With thinking off, the model responds faster but scores lower on reasoning benchmarks. With thinking on, responses include hidden reasoning tokens (2-5x longer internally) but measurably better results on hard problems.

How to run

Ollama

```
# 35B MoE (recommended for 24GB GPUs)
ollama run qwen3.5:35b-a3b
```

```
# 27B dense (for tighter cards)
ollama run qwen3.5:27b

# 122B MoE (needs 70GB+)
ollama run qwen3.5:122b-a10b
```

Ollama selects a quantization automatically based on your available memory. Tag names may vary — check the Ollama library for current tags.

llama.cpp

Download GGUF files from Unsloth (recommended quants) or lmstudio-community:

```
# Download 35B MoE Q4
huggingface-cli download unsloth/Qwen3.5-35B-A3B-GGUF \
  --local-dir Qwen3.5-35B \
  --include "*UD-Q4_K_XL*"

# Run with full GPU offload
llama-cli -m Qwen3.5-35B/Qwen3.5-35B-A3B-UD-Q4_K_XL.gguf \
  -c 8192 -ngl 999
```

Use `-ngl 999` to offload all layers to GPU. Reduce if you need to split between GPU and RAM.

LM Studio

Search for “Qwen3.5” in the model browser. Community quants are available for all three sizes. On Mac, confirm you’re using the **MLX backend** (Settings → Runtime) for best performance.

Quantization picks

Unsloth recommends their **UD-Q4_K_XL** format for the best quality-to-size ratio. Their testing shows Q3 and Q4 produce “effectively similar quality” on Qwen 3.5, so you can drop to Q3 if you need memory savings without a major quality hit.

Your VRAM	27B pick	35B-A3B pick
12 GB	Q3_K_M (tight, short context)	Too large

Your VRAM	27B pick	35B-A3B pick
16 GB	Q3_K_M (comfortable)	Too large
24 GB	Q6_K or Q8_0	Q4_K_M
32 GB	Q8_0	Q6_K
48 GB+	FP16	Q8_0

For the 122B-A10B: Q4_K_M on a 128GB Mac, Q3_K_M if you're tighter on memory.

Avoid going below Q2 on any model. Quality at IQ2 and below degrades noticeably, especially for vision and reasoning tasks.

Pick by hardware

Your setup	Recommended model	Why
RTX 3060 12GB	27B at Q3, or 9B at Q4	27B is tight. The 9B fits easily and is the better pick for 8GB cards.
RTX 4060 Ti 16GB	27B at Q3-Q4	Comfortable at Q3 with room for 8K+ context.
RTX 3090 / 4090 24GB	**35B-A3B at Q4**	The sweet spot. ~111 tok/s with room for long conversations.
RTX 5090 32GB	35B-A3B at Q6	Higher quant, more context room, ~165 tok/s.
2x RTX 3090 48GB	122B-A10B at Q3	Or 35B-A3B at Q8 for maximum quality at that size.
Mac 48-64GB	35B-A3B at Q4-Q6	Fast enough for interactive use via MLX.
Mac M4 Max 128GB	122B-A10B at Q4	Fits with headroom. Real quality step over the 35B.
Mac M3/M4 Ultra 256GB	122B-A10B at Q8	Best quality, or run 35B-A3B + another model at the same time.

The 35B-A3B on a 24GB card is the value play here. 111 tok/s generation, and benchmark scores matching much larger dense models.

Qwen 3.5 vs Qwen 3: should you switch?

If you're running Qwen3-30B-A3B, the 35B-A3B is the natural upgrade. Better benchmarks across the board, 262K context (up from 131K), and native vision.

One caveat: early llama.cpp builds show the 35B-A3B running ~35% slower than the 30B-A3B on CUDA. This is an implementation issue with the new Gated DeltaNet layers, not a fundamental problem. It will improve as llama.cpp adds optimizations. On MLX (Mac), no slowdown.

If you're running Qwen3-32B (dense) or Qwen3-14B, the 27B is worth testing. Newer architecture, better context handling, built-in vision.

Bottom line

For most people with a 24GB GPU, the 35B-A3B is the answer. 111 tok/s on an RTX 3090 at Q4, benchmark scores that match much larger models, and it fits with room for long context.

On a 16GB card, the 27B dense at Q3 is the way in. Slower, but every byte of loaded weight is active.

Mac Studio owners with 128GB should look at the 122B-A10B at Q4 — a real quality step up without needing the 397B's 192GB minimum.

Thinking mode is on by default across the lineup. Turn it off for speed, leave it on when accuracy matters. Vision and Apache 2.0 licensing come standard.

If you have the VRAM for the 35B MoE, start there.
