AI BENCH LAB

Setting Up Local Providers

Configure Ollama, LM Studio, or the built-in runtime to benchmark models on your own hardware.

Version 1.0 | March 2026

aibenchlab.com

Guides · Professional AI Model Benchmarking for Windows

Contents

Configure Ollama, LM Studio, or the built-in runtime to benchmark models on your own hardware.

1. Three Ways to Run Models Locally

AiBenchLab supports three local inference providers. You can use any combination simultaneously.

Provider	Protocol	Default Port	GPU Support	Setup Required
Built-in llama.cpp	OpenAI-compatible	8900–9999 (auto)	CUDA, Vulkan	None — ships with the app
Ollama	Ollama (dedicated)	11434	Via Ollama	Install Ollama separately
LM Studio	OpenAI-compatible	1234	Via LM Studio	Install LM Studio separately

2. Built-In llama.cpp Server

AiBenchLab includes its own llama.cpp inference server. It is fully managed by the app — no configuration, no terminal commands, no external software.

How It Works

› You select a local GGUF model (from the catalog or a file on disk).

› The app launches a llama.cpp server instance for that model on an auto-assigned port (8900–9999).

› The server exposes an OpenAI-compatible API at `http://127.0.0.1:<port>`.

› When the benchmark completes (or you unload the model), the server shuts down.

Each model gets its own server instance. The app manages the full lifecycle — start, health check, error recovery, and shutdown.

GPU Acceleration

During first-launch setup, the app downloads one of two backends:

Backend	When Selected	Download Size	GPU Compatibility
CUDA	NVIDIA GPU detected	~141 MB	NVIDIA GPUs (CUDA 13.1+)

Backend	When Selected	Download Size	GPU Compatibility
Vulkan	No NVIDIA GPU, or AMD/Intel GPU	~55 MB	AMD, Intel, NVIDIA (universal)

If you have an NVIDIA GPU, the CUDA backend is selected automatically. If no compatible GPU backend is found at runtime, the server falls back to CPU inference and logs a warning.

Server Configuration

When launching a model, the app configures:

Parameter	Default	Notes
Context size	4,096 tokens	Adjustable per model
GPU layers	Auto (all layers)	Set to 0 for CPU-only
Host	`127.0.0.1`	Localhost only — not exposed to the network

Error Recovery

The built-in server detects common failure modes and provides actionable guidance:

- **Chat template parsing error**: Retries with `--no-jinja --chat-template chatml`
- **CUDA / cuBLAS error**: Suggests reducing GPU layers or switching to Vulkan
- **Out of memory / insufficient VRAM**: Suggests a smaller model or quantization
- **Invalid GGUF magic**: Model file is corrupted — re-download recommended
- **Context size too large**: Reduce context size in model settings

System Check

Go to **Settings > Components** to run a system check. The check verifies:

- llama.cpp binary is installed and executable
- GPU backend DLLs are present (CUDA or Vulkan)
- CUDA runtime libraries (Windows: `cudart64_*.dll`, `cublas64_*.dll`)
- Loaded backends (CPU, CUDA, Vulkan) via `--version` output
- GPU device detection

The result is a pass/fail summary with details on available backends and any missing components.

---

3. Ollama

Ollama is a standalone local inference server. AiBenchLab detects it automatically on startup.

Setup

› **Install Ollama** from ollama.ai.

› **Pull a model**:

BASH

```
ollama pull llama3.2
```

› **Start the server** (if not already running):

BASH

```
ollama serve
```

› **Launch AiBenchLab.** Ollama is auto-detected at `http://localhost:11434`. Its models appear in the model selection step of the wizard.

Connection Details

Setting	Value
Endpoint	`http://localhost:11434`
API	Ollama native (`/api/tags`, `/api/show`, `/api/chat`)
Authentication	None required
Auto-detected	Yes — checked on every app launch

Model Discovery

AiBenchLab fetches your installed models via `GET /api/tags`, then queries each model's metadata via `POST /api/show` to extract:

- Context window size
- Parameter count
- Vision capability (detected from model families: `clip`, `gemma3`, `mllama`, or names containing `vl`/`vision`)

Multimodal Support

Vision-capable Ollama models (e.g., LLaVA, Gemma 3) are automatically detected and can run multimodal benchmark tests.

---

4. LM Studio

LM Studio provides a GUI for downloading and running local models with an OpenAI-compatible API.

Setup

› **Install LM Studio** from lmstudio.ai.

› **Download a model** through the LM Studio interface.

› **Load the model** and **start the local server** (LM Studio > Local Server > Start).

› **Launch AiBenchLab.** LM Studio is detected at `http://localhost:1234/v1`.

Connection Details

Setting	Value
Endpoint	`http://localhost:1234/v1`
API	OpenAI-compatible (`/models`, `/chat/completions`)
Authentication	None required for localhost
Auto-detected	Yes — health-check probed on startup

LM Studio may take a moment to become available after starting — the app confirms the connection is stable before listing it as ready. This accounts for LM Studio's model loading time.

Model Discovery

Models are fetched via the standard OpenAI `GET /models` endpoint. Embedding models are automatically filtered out.

---

5. Adding Providers Manually

If a provider isn't auto-detected (e.g., running on a non-default port), add it manually:

› Go to **Settings** in the sidebar.

› Click **+ Add Provider**.

› Select the provider template (Ollama, LM Studio, or Custom OpenAI-Compatible).

› Edit the endpoint URL if needed.

› Click **Save**, then **Test Connection**.

The test verifies connectivity, counts available models, and measures response time.

---

6. The Model Catalog

AiBenchLab includes a searchable catalog of over **51,000 HuggingFace models**, sorted by popularity. The catalog ships with a bundled seed database of the top 5,000 models for instant availability.

Browsing Models

› Click **Model Catalog** in the sidebar.

› Use **size filter buttons** (Tiny, Small, Medium, Large, XL) for quick filtering.

› Check **"Only show models that fit my hardware"** to filter by your GPU's VRAM.

› Apply presets:

• **Production Ready**: Commercial license, chat template, evaluation results

• **Commercial Safe**: Legally safe for business use

• **Air-Gapped**: Offline-capable local models

• **Quick Explore**: Minimal filters

GPU Fit Detection

The **GPU Fit** column shows whether a model fits your hardware:

Status	Meaning
Fits your GPU (green)	Model fits within available VRAM
Too large (red)	Model exceeds GPU capacity
May need offloading (amber)	Might work with partial CPU offloading
Unknown (gray)	VRAM estimate not available

GPU detection uses `nvidia-smi` on NVIDIA systems and DXGI on Windows for other GPUs.

Downloading Models

Click **Get** on any model card. The download routes through the appropriate handler:

- **GGUF models** (HuggingFace): Downloaded directly to your local models directory
- **Ollama models**: Triggers `ollama pull` if Ollama is running
- **LM Studio models**: Opens the download in LM Studio

Refreshing the Catalog

Click **Refresh Sources** and choose a sync limit:

Limit	Duration	Description
5,000	~1 min	Fast sync
10,000	~2 min	Medium sync
20,000	~5 min	Recommended default
50,000	~10 min	Full catalog

Models are synced by download count — you always get the most popular models first.

---

7. Troubleshooting

Ollama not detected

- Verify Ollama is running: `ollama list` should return your models.
- Check the port: `curl http://localhost:11434/api/tags` should return JSON.
- If using a non-default port, add the provider manually in Settings.

LM Studio not detected

- Ensure the local server is running (LM Studio > Local Server > Start).
- Verify a model is loaded — LM Studio won't respond to API calls without a model.
- Check the port: `curl http://localhost:1234/v1/models` should return JSON.

Built-in server fails to start

- Run the system check in Settings > Components.
- If CUDA errors appear, try switching to the Vulkan backend in Components.
- For "out of memory" errors, try a smaller model or reduce GPU layers.
- Check that no other application is using ports in the 8900–9999 range.

Models not appearing

- Verify the provider shows a green connection status in Settings.
- Click **Refresh** on the provider card to re-scan for models.
- For Ollama, ensure you've pulled at least one model: `ollama pull llama3.2`

Resources

- Documentation: **aibenchlab.com/docs**
- Support: **aibenchlab.com/contact**
- YouTube walkthroughs: **youtube.com/@aibenchlab**
- Email: **support@aibenchlab.com**

AiBenchLab is built by a solo founder with 40+ years of software engineering experience. Every feature is designed with the conviction that you deserve honest, transparent, and reproducible AI model evaluations. No hype. Just truth.