Hugging-Face

Ollama can run models from Hugging Face 🎉 . This integration allows you to leverage Hugging Face's vast collection of models (over 45,000 public GGUF checkpoints) with Ollama's user-friendly local execution environment. Below, I'll explain the methods, steps, and customization options based on the search results.

🛠️ Methods to Run Hugging Face Models with Ollama

There are two primary approaches to run Hugging Face models with Ollama:

1. Direct Method: For models already in GGUF format on Hugging Face.
2. Conversion Method: For models not available in GGUF format, requiring conversion using tools like llama.cpp.

🔧 Direct Method: Using Pre-Quantized GGUF Models from Hugging Face

This is the simplest method and works for models already available in GGUF format on Hugging Face.

Steps:

1. Enable Ollama Integration: · In your Hugging Face account, enable Ollama under Local Apps settings.
2. Run the Model: · On the model's Hugging Face page (e.g., bartowski/Llama-3.2-1B-Instruct-GGUF), select ollama from the "Use this model" dropdown. · Use the provided command format:

   ollama run hf.co/{username}/{repository}

   · Example:

   ollama run hf.co/bartowski/Llama-3.2-1B-Instruct-GGUF

[citation:1]

3. Specify Quantization (Optional): · If multiple quantizations are available (e.g., Q4_K_M, Q8_0, IQ3_M), specify your preferred version:

   ollama run hf.co/bartowski/Llama-3.2-3B-Instruct-GGUF:IQ3_M

   · Quantization names are case-insensitive.

4. Run Private Models: · For private GGUF models, add your Ollama SSH key to your Hugging Face account:

   cat ~/.ollama/id_ed25519.pub | pbcopy # Copy key

   · Paste the key in your Hugging Face account settings under SSH Keys.

🔄 Conversion Method: Convert Non-GGUF Models to GGUF Format

For models not available in GGUF format, use llama.cpp to convert them.

Steps:

1. Install Prerequisites: · Install huggingface_hub and clone llama.cpp:

   pip install huggingface_hub git clone https://github.com/ggerganov/llama.cpp pip install -r llama.cpp/requirements.txt

[citation:5]

2. Download the Model: · Use a script to download the model from Hugging Face (e.g., vicuna-13b-v1.5):

   from huggingface_hub import snapshot_download model_id = "lmsys/vicuna-13b-v1.5" snapshot_download(repo_id=model_id, local_dir="vicuna-hf", revision="main")

[citation:5]

3. Convert to GGUF: · Use convert.py from llama.cpp:

   python llama.cpp/convert.py vicuna-hf --outfile vicuna-13b-v1.5.gguf --outtype q8_0

   · Quantization options include f16 (full precision), q8_0 (8-bit), and q4_0 (4-bit).

4. Create a Modelfile: · Define parameters like FROM, TEMPLATE, and PARAMETER:

   # Modelfile

   FROM "./vicuna-13b-v1.5.gguf" TEMPLATE """{{ if .System }}<|system|> {{ .System }}<|end|> {{ end }} {{ if .Prompt }}<|user|> {{ .Prompt }}<|end|> {{ end }}<|assistant|> {{ .Response }}<|end|> """ PARAMETER stop "<|im_end|>" PARAMETER temperature 0.1

[citation:3][citation:8]

5. Build and Run: · Build the model:

   ollama create my-model -f Modelfile

   · Run it:

   ollama run my-model

[citation:3][citation:8]

⚙️ Customization Options

· Chat Templates: Create a template file in the repository to customize the chat format using Go templates. · System Prompts: Add a system file to define default instructions for the model. · Parameters: Create a params file (JSON format) to set sampling parameters like temperature, top_p, etc.

💻 Example Use Cases

1. Chat Completion: · Use models like microsoft/Phi-3-mini-4k-instruct-gguf for chatbot applications.
2. Embedding Models: · Run embedding models (e.g., for semantic search) by configuring them in Ollama.
3. Private Data Processing: · Run local models for data-sensitive tasks without cloud dependency.

💎 Key Considerations

· Quantization Trade-offs: Lower quantization (e.g., 4-bit) reduces resource usage but may affect accuracy. · Hardware Requirements: Larger models require more RAM/VRAM (16GB+ recommended for 7B models). · Model Compatibility: Ensure the model is supported by llama.cpp (most architectures are).

📊 Comparison of Methods

Method Pros Cons Best For Direct GGUF No conversion needed; single command Limited to existing GGUF models Quick testing; pre-quantized models Conversion Works with any Hugging Face model Requires manual steps and tools Custom models or specific versions

---

🌐 Conclusion

Ollama seamlessly integrates with Hugging Face, allowing you to run thousands of models locally. Whether you use pre-quantized GGUF models or convert others, Ollama provides a flexible environment for local AI experimentation. For further details, refer to the Hugging Face Ollama documentation or llama.cpp guide.