

TÀI LIỆU HƯỚNG DẪN

CẨM NANG CÀI ĐẶT & SỬ DỤNG

COLIBRÌ BY JUSTVUGG

Chạy Model MoE 744B–2.8T Tham Số Trên Máy Cá Nhân: Pure C • Zero Dependencies • Expert Streamed Từ Ổ đĩa

★ **GitHub Repository:** <https://github.com/JustVugg/colibri> (~28.500 Stars)

👑 **Tác giả:** Vincenzo Fornaro ([JustVugg](#))

🌐 **Trang chủ:** justvugg.github.io/colibri 💬 Discord: discord.gg/FkyrEeJR

1. COLIBRÌ LÀ GÌ & TẠI SAO ĐÁNG CHÚ Ý?

Colibrì ("chim ruồi" — **tiếng Ý**) là một inference engine mã nguồn mở giúp chạy các model MoE (Mixture-of-Experts) khổng lồ — từ **744 tỷ đến 2.8 nghìn tỷ tham số** — ngay trên phần cứng phổ thông bạn đang có, bằng cách coi ổ đĩa, RAM và VRAM là **một hệ thống bộ nhớ phân tầng duy nhất** (AI memory multitiering).

- **Ý tưởng cốt lõi — "JIT cho trọng số":** một model MoE 744B chỉ kích hoạt ~40B tham số mỗi token, và chỉ ~11GB trong số đó thay đổi giữa các token. Phần dense (~17B) luôn nằm trong RAM ở int4 (~9.9GB); 19.456 expert còn lại (~370GB) nằm trên đĩa, được nạp vào theo nhu cầu (streamed on demand).
- **Không cần GPU:** engine chạy được hoàn toàn trên CPU; GPU (CUDA, Metal, Vulkan) chỉ giúp nhanh hơn, không phải điều kiện bắt buộc cho hầu hết model.
- **Pure C, zero dependency:** engine chỉ là một file .c (vài trăm KB) cộng vài header nhỏ — không BLAS, không cần Python khi chạy (Python chỉ dùng cho bước convert model một lần và API gateway tùy chọn).
- **8 dòng model hỗ trợ:** GLM-5.2/5.3 (744B), GLM-5.3-Flash (321B, có vision), Inkling (975B), Kimi K3 (2.8T), DeepSeek V4 Flash (284B), Qwen3.8-Flash-Next (125B), Qwen3.6 (35B-A3B) và OLMoE (7B) — dùng chung giao diện coli chat / coli serve / coli web.
- **Giấy phép:** Apache 2.0. Trọng số GLM-5.2 do Z.ai phát hành theo MIT.

2. YÊU CẦU PHẦN CỨNG THEO TỪNG MODEL

Đây là điểm quan trọng nhất cần đọc kỹ trước khi chọn model — mỗi dòng model có yêu cầu đĩa/RAM/GPU rất khác nhau, không model nào bắt buộc phải có GPU.

Model	Dung lượng đĩa	RAM	GPU
OLMoE	~7GB (int8)	8GB	Không cần
GLM-5.2/5.3	~372GB	16GB tối thiểu, 24GB thoải mái	Không cần
GLM-5.3-Flash	~195GB	25GB	Không cần
Inkling	~469GB	25GB (container int4 dense)	Không cần
Kimi K3	~1.6TB	32GB+	Không cần

Model	Dung lượng đĩa	RAM	GPU
DeepSeek V4 Flash	~167GB (bản REAP 150B: ~85GB)	16GB tối thiểu, 32GB thoải mái	Tuỳ chọn — NVIDIA GTX 10 series trở lên
Qwen3.8-Flash-Next	~185.5GB (FP8)	16GB tối thiểu, 24GB thoải mái	Không hỗ trợ — chỉ CPU
Qwen3.6 (35B-A3B)	~20GB (int4-gs64)	24GB	Tuỳ chọn — 2 GPU 8GB tăng tốc 7 lần

- Tốc độ phụ thuộc chủ yếu vào ổ đĩa (vì expert được stream từ đĩa) — nên dùng SSD/NVMe nhanh; đĩa quay (HDD) sẽ rất chậm.
- Python 3 là bắt buộc để chạy launcher **coli** (bản thân engine C không cần Python).

3. HƯỚNG DẪN CÀI ĐẶT ENGINE

Cách 1 — Tải bản dựng sẵn (không cần compiler)

Hỗ trợ Linux, macOS và Windows. Tải bản đúng nền tảng từ [trang Releases](#), giải nén rồi kiểm tra:

```
mkdir colibri && tar xzf colibri-v1.8.0-linux-x86_64.tar.gz -C colibri && cd colibri
python3 coli info # engine ready ✓
```

Trong thư mục giải nén có engine (colibri, hoặc colibri.exe trên Windows), launcher coli và các script Python hỗ trợ — không cần đổi tên hay cấu hình gì, coli tự tìm engine nằm cạnh nó. Chỉ cần máy đã cài [Python 3](#).

Cách 2 — Build từ mã nguồn

Cần gcc (hoặc clang) kèm OpenMP:

```
git clone https://github.com/JustVugg/colibri && cd colibri/c
./setup.sh # kiểm tra gcc/OpenMP, build, tự test
```

setup.sh kiểm tra compiler và OpenMP, build engine và chạy self-test nhỏ. Khi thấy dòng **engine self-test: 32/32** (một số toolchain báo 30/32 hoặc 31/32 do 2 vị trí có sai số dấu phẩy động rất nhỏ — vẫn là kết quả hợp lệ) là engine đã hoạt động đúng.

Muốn có lệnh **coli** trong PATH: từ bản checkout, chạy **pip install -e .** (Đăng ký ở dạng editable install từ mã nguồn, engine vẫn nằm trong c/, không phải một wheel riêng).

Windows — build native (MSYS2)

Cài MSYS2, mở shell UCRT64, chạy:

```
pacman -S --needed mingw-w64-ucrt-x86_64-gcc make git python
```

Bản Release cho Windows kèm sẵn **coli.cmd** — double-click để chạy nhanh, hoặc gõ **coli.cmd chat --model D:\glm52_i4** từ cmd/PowerShell. Từ bản checkout mã nguồn, dùng lệnh tương đương **python coli chat --model D:\glm52_i4**. Lưu ý: các file .exe là engine, không phải launcher — chạy trực tiếp chúng sẽ không có model để nạp và tự thoát ngay.

macOS — build native

```
xcode-select --install # C compiler (clang)
brew install libomp git python # OpenMP cho multithreading
```

4. TẢI MODEL

Cần 2 thứ: chương trình (vài trăm KB, đã cài ở mục 3) và model (hàng trăm GB). Model tham chiếu chính là GLM-5.2, bản đã convert sẵn sang int4:

- **Nguồn model:** huggingface.co/mastouri/GLM-5.2-colibri-int4-g64-with-int8-mtp — dùng bản group-scaled (gs64) với đầu MTP int8, khoảng 372GB.
- **Cảnh báo quan trọng từ tác giả:** không dùng các bản mirror int4 per-row cũ (mateogrgic/..., jlmsrk/...) — README ghi rõ các bản này đo được chất lượng kém hơn ~9 điểm phần trăm và là nguyên nhân gốc của lỗi model lặp suy luận vô tận đã được báo cáo trong repo.

Hoặc tự convert từ nguồn FP8 — một lệnh có thể chạy lại (resumable), không cần đủ 756GB trống cùng lúc:

```
./coli convert --model /nvme/glm52_i4 # tải + convert theo từng shard
```

5. CHẠY MODEL

```
COLI_MODEL=/nvme/glm52_i4 ./coli chat # tự nhận diện RAM budget, cache và MTP
COLI_MODEL=/nvme/glm52_i4 ./coli plan # xem trước cách phân bổ VRAM/RAM/đĩa
COLI_MODEL=/nvme/glm52_i4 ./coli doctor # kiểm tra sẵn sàng (read-only)
COLI_MODEL=/nvme/glm52_i4 ./coli doctor --deep # kiểm tra kỹ hơn (shards/mirror)
COLI_MODEL=/nvme/glm52_i4 ./coli tune # đo và lưu cấu hình chạy nhanh nhất an toàn cho máy này
./coli web --model /nvme/glm52_i4 # API + dashboard, tự mở trình duyệt
./coli serve --model /nvme/glm52_i4 # API + dashboard, không mở trình duyệt (headless)
```

Muốn chạy model khác chỉ cần build engine tương ứng một lần rồi trở COLI_MODEL sang đúng thư mục — cú pháp dòng lệnh không đổi giữa các model:

```
make -C c glm # GLM-5.2
make -C c inkling # Inkling
make -C c kimi_k3 # Kimi K3

COLI_MODEL=/nvme/glm52_i4 ./coli chat
COLI_MODEL=/nvme/inkling_i4 ./coli chat
COLI_MODEL=/nvme/kimi_k3 ./coli chat
```

6. CHẾ ĐỘ CLUSTER NHIỀU MÁY (TUỖ CHỌN)

Cho phép nhiều máy Mac cùng chạy một model: một máy điều phối (coordinator) giữ token generation, routing và trạng thái KV, các máy worker khác thực hiện phần expert FFN đã được route tới.

```
./coli cluster coordinator --host 0.0.0.0 --port 8765
```

Trên mỗi máy worker (cần có cùng bản model đã convert):

```
./coli cluster worker --model /nvme/glm52_i4 --port 9100 \
--coordinator http://COORDINATOR:8765 --advertise-host WORKER_IP
```

Chạy coordinator kèm chế độ dò tìm tự động, hoặc chỉ định tĩnh danh sách worker qua --cluster-workers HOST:PORT,...:

```
./coli serve --model /nvme/glm52_i4 \
--cluster-coordinator http://127.0.0.1:8765
```

7. TÀI LIỆU CHUYÊN SÂU & LƯU Ý

Chủ đề	Tài liệu
Benchmark, số liệu cộng đồng	docs/benchmarks.md
Tinh chỉnh (tuning), learning cache	docs/tuning.md
Build Windows 11 native (+ CUDA DLL)	docs/windows.md
Backend CUDA, tầng expert VRAM	docs/cuda.md
Backend Vulkan (mọi GPU, kể cả AMD qua RADV)	docs/vulkan.md
Backend Metal cho Apple Silicon	docs/metal.md
API tương thích OpenAI, dashboard web	docs/api.md
Biến môi trường đầy đủ	docs/ENVIRONMENT.md

- **Ổ đĩa thứ hai (dual-SSD):** nếu có 2 SSD, đặt bản copy đầy đủ của model lên ổ thứ hai để đọc song song từ cả 2 ổ, tăng băng thông tổng — kích hoạt bằng biến COLI_MODEL_MIRROR (xem README, mục "Dual-SSD").
- **O_DIRECT trên NVMe thật:** thử bật DIRECT=1 — có thể tăng tốc decode đáng kể trên ổ có DRAM cache, nhưng tùy ổ đĩa nên cần đo thử trên máy của bạn.
- **Cẩn thận với repo/model giả mạo:** chỉ nên clone từ đúng github.com/JustVugg/colibri và chỉ tải model từ các link Hugging Face được liệt kê chính thức trong README — README của repo tự cảnh báo có những bản mirror model chất lượng kém hơn đang lưu hành, không phải do fork mã nguồn giả mạo mà do model được convert không đúng cách.

Chúc bạn nuôi "chú chim ruồi" chạy được cả những model hàng trăm tỷ tham số trên chính máy của mình!

© Kênh AI Mỗi Ngày | Hướng dẫn AI & Linux Tự Động Hóa
Nội dung kỹ thuật tổng hợp từ tài liệu chính thức của [repo github.com/JustVugg/colibri](https://github.com/JustVugg/colibri).

Tài khoản githot mới được tặng 5 xu để dùng thử công cụ AI Repo Explainer hoặc Repo Health Check.
Đăng ký: githot.io.vn/register · Tìm thêm hướng dẫn cài đặt: githot.io.vn