Sign in

ggml-org / llama.cpp Public

Notifications | Fork 16.3k | Star 101k

Code | Issues 515 | Pull requests 826 | Discussions | Actions | Projects | Wiki

master | Go to file | Code

3 people | docker : bump cuda12 t... | 277ff5f · 1 hour ago

.devops	docker : bump cuda12...	1 hour ago
.gemini	contributing: tighten ...	4 months ago
.github	docker : bump cuda12...	1 hour ago
benches	benches : add Nemotr...	3 weeks ago
ci	ci: add more binary ch...	2 hours ago
cmake	ci : add sanitizer runs f...	2 months ago
common	jinja: coerce input for ...	1 hour ago
docs	docs: Update build.m...	1 hour ago
examples	model, mtmd: fix gguf...	yesterday
ggml	ggml-zendnn : add M...	5 hours ago
gguf-py	model, mtmd: fix gguf...	yesterday
grammars	docs : document that J...	4 months ago
include	llama : refactor llama_...	2 days ago
licenses	refactor : remove libcu...	3 months ago
media	media : add transpare...	7 months ago
models	fix: gemma 4 templat...	17 hours ago
pocs	ggml : move AMX to t...	2 years ago
requirements	ci : limit requirements ...	last week
scripts	sync : ggml	yesterday
src	fix: remove stale asser...	3 hours ago
tests	jinja: coerce input for ...	1 hour ago
tools	model, mtmd: fix gguf...	yesterday
vendor	vendor : update Borin...	3 days ago
.clang-format	fix: apply clang-format...	7 months ago
.clang-tidy	clang-tidy : disable wa...	7 months ago
.dockerignore	ci : fix docker build nu...	2 years ago
.ecrc	common : Update stb...	2 years ago
.editorconfig	server: (webui) no mo...	3 days ago
.flake8	llama : move end-user...	last year
.gitattributes	server: (webui) no mo...	3 days ago
.gitignore	server: (webui) no mo...	3 days ago
.gitmodules	ggml : remove kompu...	10 months ago
.pre-commit-config.y...	convert.py : add pytho...	2 years ago
AGENTS.md	contrib : rewrite AGEN...	2 days ago
AUTHORS	authors : update (#19...	2 months ago
CLAUDE.md	contributing: tighten ...	4 months ago
CMakeLists.txt	server: Introduce LLA...	last week
CMakePresets.json	cmake : Add CMake pr...	9 months ago
CODEOWNERS	Add codeowners for s...	2 weeks ago
CONTRIBUTING.md	docs : explicit about b...	2 weeks ago
LICENSE	docs : Minor cleanups ...	2 months ago
Makefile	make : remove make i...	8 months ago
README.md	readme : clarify MODE...	last week
SECURITY.md	docs : fix broken link a...	2 months ago
build-xcframework.sh	build : remove LLAMA...	2 months ago
convert_hf_to_gguf.py	vocab: fix Gemma4 to...	6 hours ago
convert_hf_to_gguf_...	convert : add RuGPT3...	last week
convert_llama_ggml...	ci : switch from pyrigh...	2 weeks ago
convert_lora_to_gguf...	ci : switch from pyrigh...	2 weeks ago
flake.lock	flake.lock: Update (#1...	2 years ago
flake.nix	fix(nix): remove non-f...	8 months ago
mypy.ini	convert : partially reve...	2 years ago
poetry.lock	build(python): Packag...	2 years ago
pyproject.toml	gguf-py : bump sente...	2 months ago
pyrightconfig.json	ci : switch from pyrigh...	2 weeks ago
requirements.txt	`tool-call`: fix Qwen 2....	last year
ty.toml	ci : switch from pyrigh...	2 weeks ago

About

LLM inference in C/C++

ggml

- Readme
- MIT license
- Contributing
- Security policy
- Activity
- Custom properties
- 101k stars
- 665 watching
- 16.3k forks

Report repository

Releases 5,000+

b8648 Latest
1 hour ago

+ 5,712 releases

Packages 1

llama.cpp

Contributors 1,575

+ 1,561 contributors

Languages

- C++ 57.0%
- C 12.4%
- Python 7.4%
- Cuda 5.9%
- HTML 3.7%
- TypeScript 2.9%
- Other 10.7%

README | Contributing | MIT license | Security

llama.cpp

license MIT | release b8648 | Server passing

Manifesto / ggml / ops

LLM inference in C/C++

Recent API changes

- Changelog for `libllama` API
- Changelog for `llama-server` REST API

Hot topics

- **Hugging Face cache migration: models downloaded with `-hf` are now stored in the standard Hugging Face cache directory, enabling sharing with other HF tools.**
- **guide : using the new WebUI of llama.cpp**
- guide : running gpt-oss with llama.cpp
- [FEEDBACK] Better packaging for llama.cpp to support downstream consumers 🤗
- Support for the `gpt-oss` model with native MXFP4 format has been added | PR | Collaboration with NVIDIA | Comment
- Multimodal support arrived in `llama-server` : #12898 | documentation
- VS Code extension for FIM completions: https://github.com/ggml-org/llama.vscode
- Vim/Neovim plugin for FIM completions: https://github.com/ggml-org/llama.vim
- Hugging Face Inference Endpoints now support GGUF out of the box! #9669
- Hugging Face GGUF editor: discussion | tool

Quick start

Getting started with llama.cpp is straightforward. Here are several ways to install it on your machine:

- Install `llama.cpp` using brew, nix or winget
- Run with Docker - see our Docker documentation
- Download pre-built binaries from the releases page
- Build from source by cloning this repository - check out our build guide

Once installed, you'll need a model to work with. Head to the Obtaining and quantizing models section to learn more.

Example command:

```
# Use a local model file
llama-cli -m my_model.gguf

# Or download and run a model directly from Hugg
llama-cli -hf ggml-org/gemma-3-1b-it-GGUF

# Launch OpenAI-compatible API server
llama-server -hf ggml-org/gemma-3-1b-it-GGUF
```

Description

The main goal of `llama.cpp` is to enable LLM inference with minimal setup and state-of-the-art performance on a wide range of hardware - locally and in the cloud.

- Plain C/C++ implementation without any dependencies
- Apple silicon is a first-class citizen - optimized via ARM NEON, Accelerate and Metal frameworks
- AVX, AVX2, AVX512 and AMX support for x86 architectures
- RVV, ZVFH, ZFH, ZICBOP and ZIHINTPAUSE support for RISC-V architectures
- 1.5-bit, 2-bit, 3-bit, 4-bit, 5-bit, 6-bit, and 8-bit integer quantization for faster inference and reduced memory use
- Custom CUDA kernels for running LLMs on NVIDIA GPUs (support for AMD GPUs via HIP and Moore Threads GPUs via MUSA)
- Vulkan and SYCL backend support
- CPU+GPU hybrid inference to partially accelerate models larger than the total VRAM capacity

The `llama.cpp` project is the main playground for developing new features for the ggml library.

▶ Models

▶ Bindings

▶ UIs

▶ Tools

▶ Infrastructure

▶ Games

Supported backends

Backend	Target devices
Metal	Apple Silicon
BLAS	All
BLIS	All
SYCL	Intel and Nvidia GPU
OpenVINO [In Progress]	Intel CPUs, GPUs, and NPUs
MUSA	Moore Threads GPU
CUDA	Nvidia GPU
HIP	AMD GPU
ZenDNN	AMD CPU
Vulkan	GPU
CANN	Ascend NPU
OpenCL	Adreno GPU
IBM zDNN	IBM Z & LinuxONE
WebGPU [In Progress]	All
RPC	All
Hexagon [In Progress]	Snapdragon
VirtGPU	VirtGPU APIR

Obtaining and quantizing models

The Hugging Face platform hosts a number of LLMs compatible with `llama.cpp` :

- Trending
- LLaMA

You can either manually download the GGUF file or directly use any `llama.cpp` -compatible models from Hugging Face or other model hosting sites, by using this CLI argument: `-hf <user>/<model>[:quant]` . For example:

```
llama-cli -hf ggml-org/gemma-3-1b-it-GGUF
```

By default, the CLI would download from Hugging Face, you can switch to other options with the environment variable `MODEL_ENDPOINT` . The `MODEL_ENDPOINT` must point to a Hugging Face compatible API endpoint.

After downloading a model, use the CLI tools to run it locally - see below.

`llama.cpp` requires the model to be stored in the GGUF file format. Models in other data formats can be converted to GGUF using the `convert_*.py` Python scripts in this repo.

The Hugging Face platform provides a variety of online tools for converting, quantizing and hosting models with `llama.cpp` :

- Use the GGUF-my-repo space to convert to GGUF format and quantize model weights to smaller sizes
- Use the GGUF-my-LoRA space to convert LoRA adapters to GGUF format (more info: #10123)
- Use the GGUF-editor space to edit GGUF meta data in the browser (more info: #9268)
- Use the Inference Endpoints to directly host `llama.cpp` in the cloud (more info: #9669)

To learn more about model quantization, read this documentation

`llama-cli`

A CLI tool for accessing and experimenting with most of `llama.cpp`'s functionality.

- ▼ Run in conversation mode
  Models with a built-in chat template will automatically activate conversation mode. If this doesn't occur, you can manually enable it by adding `-cnv` and specifying a suitable chat template with `--chat-template NAME`

```
llama-cli -m model.gguf

# > hi, who are you?
# Hi there! I'm your helpful assistant! I'm a
#
# > what is 1+1?
# Easy peasy! The answer to 1+1 is... 2!
```

- ▶ Run in conversation mode with custom chat template
- ▶ Constrain the output with a custom grammar

`llama-server`

A lightweight, OpenAI API compatible, HTTP server for serving LLMs.

- ▼ Start a local HTTP server with default configuration on port 8080

```
llama-server -m model.gguf --port 8080

# Basic web UI can be accessed via browser: h
# Chat completion endpoint: http://localhost
```

- ▶ Support multiple-users and parallel decoding
- ▶ Enable speculative decoding
- ▶ Serve an embedding model
- ▶ Serve a reranking model
- ▶ Constrain all outputs with a grammar

`llama-perplexity`

A tool for measuring the perplexity [1] (and other quality metrics) of a model over a given text.

- ▼ Measure the perplexity over a text file

```
llama-perplexity -m model.gguf -f file.txt

# [1]15.2701,[2]5.4007,[3]5.3073,[4]6.2965,[5
# Final estimate: PPL = 5.4007 +/- 0.67339
```

- ▶ Measure KL divergence

`llama-bench`

Benchmark the performance of the inference for various parameters.

- ▼ Run default benchmark

```
llama-bench -m model.gguf

# Output:
# | model                |       size |     pa
# | ------------------- | ---------: | -----
# | qwen2 1.5B Q4_0     | 885.97 MiB |      1
# | qwen2 1.5B Q4_0     | 885.97 MiB |      1
#
# build: 3e0ba0e60 (4229)
```

`llama-simple`

A minimal example for implementing apps with `llama.cpp`. Useful for developers.

- ▶ Basic text completion

Contributing

- Contributors can open PRs
- Collaborators will be invited based on contributions
- Maintainers can push to branches in the `llama.cpp` repo and merge PRs into the `master` branch
- Any help with managing issues, PRs and projects is very appreciated!
- See good first issues for tasks suitable for first contributions
- Read the CONTRIBUTING.md for more information
- Make sure to read this: Inference at the edge
- A bit of backstory for those who are interested: Changelog podcast

Other documentation

- cli
- completion
- server
- GBNF grammars

Development documentation

- How to build
- Running on Docker
- Build on Android
- Performance troubleshooting
- GGML tips & tricks

Seminal papers and background on the models

If your issue is with model generation quality, then please at least scan the following links and papers to understand the limitations of LLaMA models. This is especially important when choosing an appropriate model size and appreciating both the significant and subtle differences between LLaMA models and ChatGPT:

- LLaMA:
  - Introducing LLaMA: A foundational, 65-billion-parameter large language model
  - LLaMA: Open and Efficient Foundation Language Models
- GPT-3
  - Language Models are Few-Shot Learners
- GPT-3.5 / InstructGPT / ChatGPT:
  - Aligning language models to follow instructions
  - Training language models to follow instructions with human feedback

XCFramework

The XCFramework is a precompiled version of the library for iOS, visionOS, tvOS, and macOS. It can be used in Swift projects without the need to compile the library from source. For example:

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum v

import PackageDescription

let package = Package(
    name: "MyLlamaPackage",
    targets: [
        .executableTarget(
            name: "MyLlamaPackage",
            dependencies: [
                "LlamaFramework"
            ]),
        .binaryTarget(
```

```
            name: "LlamaFramework",
            url: "https://github.com/ggml-org/lla
            checksum: "c19be78b5f00d8d29a25da410
        )
    ]
)
```

The above example is using an intermediate build `b5046` of the library. This can be modified to use a different version by changing the URL and checksum.

Completions

Command-line completion is available for some environments.

Bash Completion

```
$ build/bin/llama-cli --completion-bash > ~/.llam
$ source ~/.llama-completion.bash
```

Optionally this can be added to your `.bashrc` or `.bash_profile` to load it automatically. For example:

```
$ echo "source ~/.llama-completion.bash" >> ~/.ba
```

Dependencies

- yhirose/cpp-httplib - Single-header HTTP server, used by `llama-server` - MIT license
- stb-image - Single-header image format decoder, used by multimodal subsystem - Public domain
- nlohmann/json - Single-header JSON library, used by various tools/examples - MIT License
- miniaudio.h - Single-header audio format decoder, used by multimodal subsystem - Public domain
- subprocess.h - Single-header process launching solution for C and C++ - Public domain

1. https://huggingface.co/docs/transformers/perplexity ↩

© 2026 GitHub, Inc.