## NAME

llamafile — large language model runner

## SYNOPSIS

**`llamafile`** [`--server`] [flags...] `-m` *`model.gguf`* [`--mmproj` *`vision.gguf`*]
**`llamafile`** [`--cli`] [flags...] `-m` *`model.gguf`* `-p` *`prompt`*
**`llamafile`** [`--cli`] [flags...] `-m` *`model.gguf`* `--mmproj` *`vision.gguf`* `--image` *`graphic.png`* `-p` *`prompt`*

## DESCRIPTION

**`llamafile`** is a large language model tool. It has use cases such as:

- Code completion
- Prose composition
- Chatbot that passes the Turing test
- Text/image summarization and analysis

## OPTIONS

The following options are available:

`--version`
Print version and exit.

`-h`, `--help`
Show help message and exit.

`--cli`
Puts program in command line interface mode. This flag is implied when a prompt is supplied using either the `-p` or `-f` flags.

`--server`
Puts program in server mode. This will launch an HTTP server on a local port. This server has both a web UI and an OpenAI API compatible completions endpoint. When the server is run on a desk system, a tab browser tab will be launched automatically that displays the web UI. This `--server` flag is implied if no prompt is specified, i.e. neither the `-p` or `-f` flags are passed.

`-m` *`FNAME`*, `--model` *`FNAME`*
Model path in the GGUF file format.

Default: *models/7B/ggml-model-f16.gguf*

`--mmproj` *`FNAME`*
Specifies path of the LLaVA vision model in the GGUF file format. If this flag is supplied, then the `--model` and `--image` flags should also be supplied.

`-s` *`SEED`*, `--seed` *`SEED`*
Random Number Generator (RNG) seed. A random seed is used if this is less than zero.

Default: -1

`-t` *`N`*, `--threads` *`N`*
Number of threads to use during generation.

Default: $(nproc)/2

`-tb` *`N`*, `--threads-batch` *`N`*
Set the number of threads to use during batch and prompt processing. In some systems, it is beneficial to use a higher number of threads during batch processing than during generation. If not specified, the number of threads used for batch processing will be the same as the number of threads used for generation.

Default: Same as `--threads`

`−td` *N*, `−−threads-draft` *N*

Number of threads to use during generation.

Default: Same as `−−threads`

`−tbd` *N*, `−−threads-batch-draft` *N*

Number of threads to use during batch and prompt processing.

Default: Same as `−−threads-draft`

`−−in-prefix-bos`

Prefix BOS to user inputs, preceding the `−−in-prefix` string.

`−−in-prefix` *STRING*

This flag is used to add a prefix to your input, primarily, this is used to insert a space after the reverse prompt. Here's an example of how to use the `−−in-prefix` flag in conjunction with the `−−reverse-prompt` flag:

```
./main -r "User:" --in-prefix " "
```

Default: empty

`−−in-suffix` *STRING*

This flag is used to add a suffix after your input. This is useful for adding an "Assistant:" prompt after the user's input. It's added after the new-line character (\n) that's automatically added to the end of the user's input. Here's an example of how to use the `−−in-suffix` flag in conjunction with the `−−reverse-prompt` flag:

```
./main -r "User:" --in-prefix " " --in-suffix "Assistant:"
```

Default: empty

`−n` *N*, `−−n-predict` *N*

Number of tokens to predict.

- -1 = infinity
- -2 = until context filled

Default: -1

`−c` *N*, `−−ctx-size` *N*

Set the size of the prompt context. A larger context size helps the model to better comprehend and generate responses for longer input or conversations. The LLaMA models were built with a context of 2048, which yields the best results on longer input / inference.

- 0 = loaded automatically from model

Default: 512

`−b` *N*, `−−batch-size` *N*

Batch size for prompt processing.

Default: 512

`−−top-k` *N*

Top-k sampling.

- 0 = disabled

Default: 40

`−−top-p` *N*

Top-p sampling.

- 1.0 = disabled

Default: 0.9

`--min-p` *N*
Min-p sampling.

- 0.0 = disabled

Default: 0.1

`--tfs` *N*
Tail free sampling, parameter z.

- 1.0 = disabled

Default: 1.0

`--typical` *N*
Locally typical sampling, parameter p.

- 1.0 = disabled

Default: 1.0

`--repeat-last-n` *N*
Last n tokens to consider for penalize.

- 0 = disabled
- -1 = ctx_size

Default: 64

`--repeat-penalty` *N*
Penalize repeat sequence of tokens.

- 1.0 = disabled

Default: 1.1

`--presence-penalty` *N*
Repeat alpha presence penalty.

- 0.0 = disabled

Default: 0.0

`--frequency-penalty` *N*
Repeat alpha frequency penalty.

- 0.0 = disabled

Default: 0.0

`--mirostat` *N*
Use Mirostat sampling. Top K, Nucleus, Tail Free and Locally Typical samplers are ignored if used..

- 0 = disabled
- 1 = Mirostat
- 2 = Mirostat 2.0

Default: 0

`--mirostat-lr` *N*
Mirostat learning rate, parameter eta.

Default: 0.1

`--mirostat-ent` *N*
: Mirostat target entropy, parameter tau.

  Default: 5.0

`-l` *TOKEN_ID(+/-)BIAS*, `--logit-bias` *TOKEN_ID(+/-)BIAS*
: Modifies the likelihood of token appearing in the completion, i.e. `--logit-bias` *15043+1* to increase likelihood of token *' Hello'*, or `--logit-bias` *15043-1* to decrease likelihood of token *' Hello'*.

`-md` *FNAME*, `--model-draft` *FNAME*
: Draft model for speculative decoding.

  Default: *models/7B/ggml-model-f16.gguf*

`--cfg-negative-prompt` *PROMPT*
: Negative prompt to use for guidance..

  Default: empty

`--cfg-negative-prompt-file` *FNAME*
: Negative prompt file to use for guidance.

  Default: empty

`--cfg-scale` *N*
: Strength of guidance.

  - 1.0 = disable

  Default: 1.0

`--rope-scaling` *{none,linear,yarn}*
: RoPE frequency scaling method, defaults to linear unless specified by the model

`--rope-scale` *N*
: RoPE context scaling factor, expands context by a factor of *N* where *N* is the linear scaling factor used by the fine-tuned model. Some fine-tuned models have extended the context length by scaling RoPE. For example, if the original pre-trained model have a context length (max sequence length) of 4096 (4k) and the fine-tuned model have 32k. That is a scaling factor of 8, and should work by setting the above `--ctx-size` to 32768 (32k) and `--rope-scale` to 8.

`--rope-freq-base` *N*
: RoPE base frequency, used by NTK-aware scaling.

  Default: loaded from model

`--rope-freq-scale` *N*
: RoPE frequency scaling factor, expands context by a factor of 1/N

`--yarn-orig-ctx` *N*
: YaRN: original context size of model.

  Default: 0 = model training context size

`--yarn-ext-factor` *N*
: YaRN: extrapolation mix factor.

  - 0.0 = full interpolation

  Default: 1.0

`--yarn-attn-factor` *N*
: YaRN: scale sqrt(t) or attention magnitude.

Default: 1.0

`--yarn-beta-slow` *N*
YaRN: high correction dim or alpha.

Default: 1.0

`--yarn-beta-fast` *N*
YaRN: low correction dim or beta.

Default: 32.0

`--ignore-eos`
Ignore end of stream token and continue generating (implies `--logit-bias` *2-inf*)

`--no-penalize-nl`
Do not penalize newline token.

`--temp` *N*
Temperature.

Default: 0.8

`--logits-all`
Return logits for all tokens in the batch.

Default: disabled

`--hellaswag`
Compute HellaSwag score over random tasks from datafile supplied with -f

`--hellaswag-tasks` *N*
Number of tasks to use when computing the HellaSwag score.

Default: 400

`--keep` *N*
This flag allows users to retain the original prompt when the model runs out of context, ensuring a connection to the initial instruction or conversation topic is maintained, where *N* is the number of tokens from the initial prompt to retain when the model resets its internal context.

- 0 = no tokens are kept from initial prompt
- -1 = retain all tokens from initial prompt

Default: 0

`--draft` *N*
Number of tokens to draft for speculative decoding.

Default: 16

`--chunks` *N*
Max number of chunks to process.

- -1 = all

Default: -1

`-ns` *N*, `--sequences` *N*
Number of sequences to decode.

Default: 1

`-pa` *N*, `--p-accept` *N*
speculative decoding accept probability.

Default: 0.5

`-ps` *N*, `--p-split` *N*
Speculative decoding split probability.

Default: 0.1

`--mlock`
Force system to keep model in RAM rather than swapping or compressing.

`--no-mmap`
Do not memory-map model (slower load but may reduce pageouts if not using mlock).

`--numa`
Attempt optimizations that help on some NUMA systems if run without this previously, it is recommended to drop the system page cache before using this. See https://github.com/ggerganov/llama.cpp/issues/1437.

`--recompile`
Force GPU support to be recompiled at runtime if possible.

`--nocompile`
Never compile GPU support at runtime.

If the appropriate DSO file already exists under *~/.llamafile/* then it'll be linked as-is without question. If a prebuilt DSO is present in the PKZIP content of the executable, then it'll be extracted and linked if possible. Otherwise, **`llamafile`** will skip any attempt to compile GPU support and simply fall back to using CPU inference.

`--gpu` *GPU*
Specifies which brand of GPU should be used. Valid choices are:

- *AUTO*: Use any GPU if possible, otherwise fall back to CPU inference (default)
- *APPLE*: Use Apple Metal GPU. This is only available on MacOS ARM64. If Metal could not be used for any reason, then a fatal error will be raised.
- *AMD*: Use AMD GPUs. The AMD HIP ROCm SDK should be installed in which case we assume the HIP_PATH environment variable has been defined. The set of gfx microarchitectures needed to run on the host machine is determined automatically based on the output of the hipInfo command. On Windows, **`llamafile`** release binaries are distributed with a tinyBLAS DLL so it'll work out of the box without requiring the HIP SDK to be installed. However, tinyBLAS is slower than rocBLAS for batch and image processing, so it's recommended that the SDK be installed anyway. If an AMD GPU could not be used for any reason, then a fatal error will be raised.
- *NVIDIA*: Use NVIDIA GPUs. If an NVIDIA GPU could not be used for any reason, a fatal error will be raised. On Windows, NVIDIA GPU support will use our tinyBLAS library, since it works on stock Windows installs. However, tinyBLAS goes slower for batch and image processing. It's possible to use NVIDIA's closed-source cuBLAS library instead. To do that, both MSVC and CUDA need to be installed and the **`llamafile`** command should be run once from the x64 MSVC command prompt with the `--recompile` flag passed. The GGML library will then be compiled and saved to *~/.llamafile/* so the special process only needs to happen a single time.
- *DISABLE*: Never use GPU and instead use CPU inference. This setting is implied by `-ngl` *0*.

`-ngl` *N*, `--n-gpu-layers` *N*
Number of layers to store in VRAM.

`-ngld` *N*, `--n-gpu-layers-draft` *N*
: Number of layers to store in VRAM for the draft model.

`-sm` *SPLIT_MODE*, `--split-mode` *SPLIT_MODE*
: How to split the model across multiple GPUs, one of:
- none: use one GPU only
- layer (default): split layers and KV across GPUs
- row: split rows across GPUs

`-ts` *SPLIT*, `--tensor-split` *SPLIT*
: When using multiple GPUs this option controls how large tensors should be split across all GPUs. *SPLIT* is a comma-separated list of non-negative values that assigns the proportion of data that each GPU should get in order. For example, "3,2" will assign 60% of the data to GPU 0 and 40% to GPU 1. By default the data is split in proportion to VRAM but this may not be optimal for performance. Requires cuBLAS. How to split tensors across multiple GPUs, comma-separated list of proportions, e.g. 3,1

`-mg` *i*, `--main-gpu` *i*
: The GPU to use for scratch and small tensors.

`-nommq`, `--no-mul-mat-q`
: Use cuBLAS instead of custom mul_mat_q CUDA kernels. Not recommended since this is both slower and uses more VRAM.

`--verbose-prompt`
: Print prompt before generation.

`--simple-io`
: Use basic IO for better compatibility in subprocesses and limited consoles.

`--lora` *FNAME*
: Apply LoRA adapter (implies `--no-mmap`)

`--lora-scaled` *FNAME S*
: Apply LoRA adapter with user defined scaling S (implies `--no-mmap`)

`--lora-base` *FNAME*
: Optional model to use as a base for the layers modified by the LoRA adapter

`--unsecure`
: Disables pledge() sandboxing on Linux and OpenBSD.

`--samplers`
: Samplers that will be used for generation in the order, separated by semicolon, for example: top_k;tfs;typical;top_p;min_p;temp

`--samplers-seq`
: Simplified sequence for samplers that will be used.

`-cml`, `--chatml`
: Run in chatml mode (use with ChatML-compatible models)

`-dkvc`, `--dump-kv-cache`
: Verbose print of the KV cache.

`-nkvo`, `--no-kv-offload`
: Disable KV offload.

`-ctk` *TYPE*, `--cache-type-k` *TYPE*
: KV cache data type for K.

`−ctv` *TYPE*, `−−cache-type-v` *TYPE*
KV cache data type for V.

`−gan` *N*, `−−grp-attn-n` *N*
Group-attention factor.

Default: 1

`−gaw` *N*, `−−grp-attn-w` *N*
Group-attention width.

Default: 512

`−bf` *FNAME*, `−−binary-file` *FNAME*
Binary file containing multiple choice tasks.

`−−winogrande`
Compute Winogrande score over random tasks from datafile supplied by the `−f` flag.

`−−winogrande-tasks` *N*
Number of tasks to use when computing the Winogrande score.

Default: 0

`−−multiple-choice`
Compute multiple choice score over random tasks from datafile supplied by the `−f` flag.

`−−multiple-choice-tasks` *N*
Number of tasks to use when computing the multiple choice score.

Default: 0

`−−kl-divergence`
Computes KL-divergence to logits provided via the `−−kl-divergence-base` flag.

`−−save-all-logits` *FNAME*, `−−kl-divergence-base` *FNAME*
Save logits to filename.

`−ptc` *N*, `−−print-token-count` *N*
Print token count every *N* tokens.

Default: -1

`−−pooling` *KIND*
Specifies pooling type for embeddings. This may be one of:

- none
- mean
- cls

The model default is used if unspecified.

## CLI OPTIONS

The following options may be specified when **`llamafile`** is running in `−−cli` mode.

`−e`, `−−escape`
Process prompt escapes sequences (\n, \r, \t, \´, \", \\)

`−p` *STRING*, `−−prompt` *STRING*
Prompt to start text generation. Your LLM works by auto-completing this text. For example:

```
llamafile -m model.gguf -p "four score and"
```

Stands a pretty good chance of printing Lincoln's Gettysburg Address. Prompts can take on a structured format too. Depending on how your model was trained, it may specify in its docs an instruction notation. With some models that might be:

```
llamafile -p "[INST]Summarize this: $(cat file)[/INST]"
```

In most cases, simply colons and newlines will work too:

```
llamafile -e -p "User: What is best in life?\nAssistant:"
```

`−f` *FNAME*, `−−file` *FNAME*
: Prompt file to start generation.

`−−grammar` *GRAMMAR*
: BNF-like grammar to constrain which tokens may be selected when generating text. For example, the grammar:

```
root ::= "yes" | "no"
```

will force the LLM to only output yes or no before exiting. This is useful for shell scripts when the `−−no-display-prompt` flag is also supplied.

`−−grammar-file` *FNAME*
: File to read grammar from.

`−−fast`
: Put llamafile into fast math mode. This disables algorithms that reduce floating point rounding, e.g. Kahan summation, and certain functions like expf() will be vectorized but handle underflows less gracefully. It's unspecified whether llamafile runs in fast or precise math mode when neither flag is specified.

`−−precise`
: Put llamafile into precise math mode. This enables algorithms that reduce floating point rounding, e.g. Kahan summation, and certain functions like expf() will always handle subnormals correctly. It's unspecified whether llamafile runs in fast or precise math mode when neither flag is specified.

`−−trap`
: Put llamafile into math trapping mode. When floating point exceptions occur, such as NaNs, overflow, and divide by zero, llamafile will print a warning to the console. This warning will include a C++ backtrace the first time an exception is trapped. The op graph will also be dumped to a file, and llamafile will report the specific op where the exception occurred. This is useful for troubleshooting when reporting issues. USing this feature will disable sandboxing. Math trapping is only possible if your CPU supports it. That is generally the case on AMD64, however it's less common on ARM64.

`−−prompt-cache` *FNAME*
: File to cache prompt state for faster startup.

  Default: none

`−fa` *FNAME*, `−−flash-attn`
: Enable Flash Attention. This is a mathematical shortcut that can speed up inference for certain models. This feature is still under active development.

`−−prompt-cache-all`
: If specified, saves user input and generations to cache as well. Not supported with `−−interactive` or other interactive options.

`−−prompt-cache-ro`
: If specified, uses the prompt cache but does not update it.

`−−random-prompt`
: Start with a randomized prompt.

`--image` *`IMAGE_FILE`*
: Path to an image file. This should be used with multimodal models. Alternatively, it's possible to embed an image directly into the prompt instead; in which case, it must be base64 encoded into an HTML img tag URL with the image/jpeg MIME type. See also the `--mmproj` flag for supplying the vision model.

`-i`, `--interactive`
: Run the program in interactive mode, allowing users to engage in real-time conversations or provide specific instructions to the model.

`--interactive-first`
: Run the program in interactive mode and immediately wait for user input before starting the text generation.

`-ins`, `--instruct`
: Run the program in instruction mode, which is specifically designed to work with Alpaca models that excel in completing tasks based on user instructions.

: Technical details: The user's input is internally prefixed with the reverse prompt (or "### Instruction:" as the default), and followed by "### Response:" (except if you just press Return without any input, to keep generating a longer response).

: By understanding and utilizing these interaction options, you can create engaging and dynamic experiences with the LLaMA models, tailoring the text generation process to your specific needs.

`-r` *`PROMPT`*, `--reverse-prompt` *`PROMPT`*
: Specify one or multiple reverse prompts to pause text generation and switch to interactive mode. For example, `-r` *`"User:"`* can be used to jump back into the conversation whenever it's the user's turn to speak. This helps create a more interactive and conversational experience. However, the reverse prompt doesn't work when it ends with a space. To overcome this limitation, you can use the `--in-prefix` flag to add a space or an y other characters after the reverse prompt.

`--color`
: Enable colorized output to differentiate visually distinguishing between prompts, user input, and generated text.

`--no-display-prompt`, `--silent-prompt`
: Don't echo the prompt itself to standard output.

`--keep` *`N`*
: Specifies number of tokens to keep from the initial prompt. The default is -1 which means all tokens.

`--multiline-input`
: Allows you to write or paste multiple lines without ending each in '\'.

`--cont-batching`
: Enables continuous batching, a.k.a. dynamic batching. is -1 which means all tokens.

`--embedding`
: In CLI mode, the embedding flag may be use to print embeddings to standard output. By default, embeddings are computed over a whole prompt. However the `--multiline` flag may be passed, to have a separate embeddings array computed for each line of text in the prompt. In multiline mode, each embedding array will be printed on its own line to standard output, where individual floats are separated by space. If both the `--multiline-input` and `--interactive` flags are passed, then a pretty-printed summary of embeddings along with a cosine similarity matrix will be printed to the terminal.

## SERVER OPTIONS

The following options may be specified when **`llamafile`** is running in `--server` mode.

`--port` *`PORT`*
: Port to listen

  Default: 8080

`--host` *`IPADDR`*
: IP address to listen.

  Default: 127.0.0.1

`-to` *`N`*, `--timeout` *`N`*
: Server read/write timeout in seconds.

  Default: 600

`-np` *`N`*, `--parallel` *`N`*
: Number of slots for process requests.

  Default: 1

`-cb`, `--cont-batching`
: Enable continuous batching (a.k.a dynamic batching).

  Default: disabled

`-spf` *`FNAME`*, `--system-prompt-file` *`FNAME`*
: Set a file to load a system prompt (initial prompt of all slots), this is useful for chat applications.

`-a` *`ALIAS`*, `--alias` *`ALIAS`*
: Set an alias for the model. This will be added as the *`model`* field in completion responses.

`--path` *`PUBLIC_PATH`*
: Path from which to serve static files.

  Default: */zip/llama.cpp/server/public*

`--nobrowser`
: Do not attempt to open a web browser tab at startup.

`-gan` *`N`*, `--grp-attn-n` *`N`*
: Set the group attention factor to extend context size through self-extend. The default value is *`1`* which means disabled. This flag is used together with `--grp-attn-w`.

`-gaw` *`N`*, `--grp-attn-w` *`N`*
: Set the group attention width to extend context size through self-extend. The default value is *`512`*. This flag is used together with`--grp-attn-n`.

## LOG OPTIONS

The following log options are available:

`-ld` *`LOGDIR`*, `--logdir` *`LOGDIR`*
: Path under which to save YAML logs (no logging if unset)

`--log-test`
: Run simple logging test

`--log-disable`
: Disable trace logs

`--log-enable`
: Enable trace logs

`−−log-file`
: Specify a log filename (without extension)

`−−log-new`
: Create a separate new log file on start. Each log file will have unique name: `<`*`name`*`>.<`*`ID`*`>.log`

`−−log-append`
: Don't truncate the old log file.

## EXAMPLES

Here's an example of how to run llama.cpp's built-in HTTP server. This example uses LLaVA v1.5-7B, a multimodal LLM that works with llama.cpp's recently-added support for image inputs.

```
llamafile \
  -m llava-v1.5-7b-Q8_0.gguf \
  --mmproj llava-v1.5-7b-mmproj-Q8_0.gguf \
  --host 0.0.0.0
```

Here's an example of how to generate code for a libc function using the llama.cpp command line interface, utilizing WizardCoder-Python-13B weights:

```
llamafile \
  -m wizardcoder-python-13b-v1.0.Q8_0.gguf --temp 0 -r '}\n' -r '```\n' \
  -e -p '```c\nvoid *memcpy(void *dst, const void *src, size_t size) {\n'
```

Here's a similar example that instead utilizes Mistral-7B-Instruct weights for prose composition:

```
llamafile \
  -m mistral-7b-instruct-v0.2.Q5_K_M.gguf \
  -p '[INST]Write a story about llamas[/INST]'
```

Here's an example of how llamafile can be used as an interactive chatbot that lets you query knowledge contained in training data:

```
llamafile -m llama-65b-Q5_K.gguf -p '
The following is a conversation between a Researcher and their helpful AI
assistant Digital Athena which is a large language model trained on the
sum of human knowledge.
Researcher: Good morning.
Digital Athena: How can I help you today?
Researcher:' --interactive --color --batch_size 1024 --ctx_size 4096 \
--keep -1 --temp 0 --mirostat 2 --in-prefix ' ' --interactive-first \
--in-suffix 'Digital Athena:' --reverse-prompt 'Researcher:'
```

Here's an example of how you can use llamafile to summarize HTML URLs:

```
(
  echo '[INST]Summarize the following text:'
  links -codepage utf-8 \
        -force-html \
        -width 500 \
        -dump https://www.poetryfoundation.org/poems/48860/the-raven |
    sed 's/   */ /g'
  echo '[/INST]'
) | llamafile \
      -m mistral-7b-instruct-v0.2.Q5_K_M.gguf \
      -f /dev/stdin \
      -c 0 \
      --temp 0 \
      -n 500 \
```

```
          --no-display-prompt 2>/dev/null
```

Here's how you can use llamafile to describe a jpg/png/gif/bmp image:

```
llamafile --temp 0 \
  --image lemurs.jpg \
  -m llava-v1.5-7b-Q4_K.gguf \
  --mmproj llava-v1.5-7b-mmproj-Q4_0.gguf \
  -e -p '### User: What do you see?\n### Assistant: ' \
  --no-display-prompt 2>/dev/null
```

If you wanted to write a script to rename all your image files, you could use the following command to generate a safe filename:

```
llamafile --temp 0 \
    --image ~/Pictures/lemurs.jpg \
    -m llava-v1.5-7b-Q4_K.gguf \
    --mmproj llava-v1.5-7b-mmproj-Q4_0.gguf \
    --grammar 'root ::= [a-z]+ (" " [a-z]+)+' \
    -e -p '### User: The image has...\n### Assistant: ' \
    --no-display-prompt 2>/dev/null |
  sed -e's/ /_/g' -e's/$/.jpg/'
three_baby_lemurs_on_the_back_of_an_adult_lemur.jpg
```

Here's an example of how to make an API request to the OpenAI API compatible completions endpoint when your **`llamafile`** is running in the background in `--server` mode.

```
curl -s http://localhost:8080/v1/chat/completions \
     -H "Content-Type: application/json" -d '{
  "model": "gpt-3.5-turbo",
  "stream": true,
  "messages": [
    {
      "role": "system",
      "content": "You are a poetic assistant."
    },
    {
      "role": "user",
      "content": "Compose a poem that explains FORTRAN."
    }
  ]
}' | python3 -c '
import json
import sys
json.dump(json.load(sys.stdin), sys.stdout, indent=2)
print()
```

## PROTIP

The `-ngl` *35* flag needs to be passed in order to use GPUs made by NVIDIA and AMD. It's not enabled by default since it sometimes needs to be tuned based on the system hardware and model architecture, in order to achieve optimal performance, and avoid compromising a shared display.

## SEE ALSO

*llamafile-quantize*(1), *llamafile-perplexity*(1), *llava-quantize*(1), *zipalign*(1), *unzip*(1)