**Technical - Large language models (LLMs)**

For the theory about NLP & Transformers, embeddings, take a look at the previous document

Large Scale Language Models (LLMs) are natural language processing (NLP) models that have been trained on extensive datasets to enhance their language processing abilities. These models have the capacity to analyze and comprehend textual data to accomplish multiple tasks like language translation, sentiment analysis and content generation.

Some of the well-known examples of large language models include OpenAI's ChatGPT, Microsoft's Turing NLG, Deepmind's Gopher and

Chinchilla, Google's T5 and MT5, and Baidu's Ernie 3.0, all of which are trained on billions of tokens.

# Read about Local LLMs:

- Getting started : <https://www.reddit.com/r/LocalLLaMA/wiki/index/>

- Install guide : <https://www.reddit.com/r/LocalLLaMA/wiki/guide/>

- Models : <https://www.reddit.com/r/LocalLLaMA/wiki/models/>

- Datasets & Projects: <https://www.reddit.com/r/LocalLLaMA/wiki/communityprojects/>

- How to set up guide : <https://www.reddit.com/r/LocalLLaMA/comments/11o6o3f/how_to_install_llama_8bit_and_4bit/>

# Available models & benchmarks :

Here is a list of Large language models :

\* (LLMs list) [LocalLLaMA/wiki/models/ & requirements](https://www.reddit.com/r/LocalLLaMA/wiki/models/)

\* (LLMs list) Download quantised models [TheBloke Huggingface](https://huggingface.co/TheBloke)

\* (LLMs list) [models requirements](https://www.reddit.com/r/LocalLLaMA/wiki/models/)

- (LLMs list Commercial Use) [eugeneyan/open-llms: A list of open LLMs available for commercial use. (github.com)](https://github.com/eugeneyan/open-llms)

- (LLMs list) [awesome-marketing-datascience/llm-model-list](https://github.com/underlines/awesome-marketing-datascience/blob/master/llm-model-list.md)

- [(LLMs list) [imaurer/awesome-decentralized-llm](https://github.com/imaurer/awesome-decentralized-llm)

- (Leaderboard) [Open LLM Leaderboard - a Hugging Face Space by HuggingFaceH4](https://huggingface.co/spaces/HuggingFaceH4/open\_llm\_leaderboard)

- [Large language model - Wikipedia](https://en.wikipedia.org/wiki/Large\_language\_model#List\_of\_large\_language\_models)

# Technical Questions

Some information here might be outdated due to the fast development and optimization of most LLMs libraries.

## Libraries & Frameworks :

To run LLMs locally you can start follow this guide

- First, Text Generation webui : <https://github.com/oobabooga/text-generation-webui> , install it and use it. There are even installers for Windows, WSL & Linux installers : <https://github.com/oobabooga/one-click-installers>

- If you want to master LLMs, every feature in Text generation webui is good to learn. For example, what is exllama, what are the different parameters ...

- Follow LocalLLaMA subreddit every day

## Building apps:

Get inspired from these libraries :

* Langchain
* Text generation web UI : always take a look at updates & requirements to discover the latest libraries they are using for optimization (LLamaCPP, ExllamaV2)
  + Text-generation-webui is a gradio web UI for running Large Language Models like GPT-J 6B, OPT, GALACTICA, LLaMA,Pygmalion and many, many more! Whisper and TTS included, allowing you to speak to your characters and have them speak back too.
  + You can check some youtube videos to show you how to install it.
* Pydantic
* LlamaCpp (CPU/GPU)
* standalon C/C++ : <https://github.com/ggerganov/llama.cpp>
* python module : <https://github.com/abetlen/llama-cpp-python>
* exllama (GPU Only):
  + standalon C/C++ : <https://github.com/turboderp/exllama>
  + python module (pip installable) : <https://github.com/jllllll/exllama>
* exllamaV2 (GPU Only):
  + standalone : <https://github.com/turboderp/exllamav2>
  + python module (pip installable) : https://github.com/jllllll/exllamav2
* Take a look at text-generation-webui to discover new libraries, and requirements.txt in their repo see which repositories they are installing.
* Libraries I stared : <https://github.com/stars/AmineDjeghri/lists/nlp>

## Other tools

**- How to create installers for you app ?** Check the Installers of text generation webui to learn some tips

- label studio for annotation

- FastAPI

- Install a cuda/torch environement easily : <https://github.com/AmineDjeghri/AwesomeWindows11/blob/master/cuda_pytorch_install.md>

You can build lot of applications, one of the most interesting ones is : RAG. With langchain, it's very easy : llamacpp, langchain, chroma db

## Questions

### How to use text generation web ui?

**Docs :**

[text-generation-webui/docs at main · oobabooga/text-generation-webui (github.com)](https://github.com/oobabooga/text-generation-webui/tree/main/docs)

**For general users (easy) :**

* Download one of the installers : <https://github.com/oobabooga/text-generation-webui#one-click-installers>
* Download executable of a [release ](https://github.com/oobabooga/text-generation-webui/releases)then run the start\_windows.bat or linux to install it. Run again start\\_windows.bat or linux after the installation to run the app.
* You can change `CMD\_FLAGS` inside `webui.py`

**For devs (advanced)**

* link : <https://github.com/oobabooga/text-generation-webui#0-install-conda>

**If it’s complicated for you, check a recent video on Youtube**

### GGML/GGUF or GPTQ ?

new : GGUF will replace GGML models (September 2023) (GGML will be outdated)

- GGML files are for CPU + GPU inference using [llama.cpp ](<https://github.com/ggerganov/llama.cpp>

)and libraries and UIs

which support this format

- GPTQ files are for GPU inference only that needs GPTQ or AutoGPTQ to run.

- Simple rule of thumb: If you can fit the entire model in VRAM + context then GPTQ is going to be significantly faster.

If not, then GGUF is faster to significantly faster depending how much layers you have to offload.

- GGUF or GPTQ : [discussion](<https://huggingface.co/TheBloke/wizardLM-7B-GGML/discussions/3#644ad9c0cb45734dfd482b11>)

- [How do I create a GGUF model file? (secondstate.io)](https://www.secondstate.io/articles/convert-pytorch-to-gguf/)

- 1.GGML models : require llama.cpp

- 2.GPTQ models : require llama-for-gptq or AutoGPTQ or exllama (3rd one the the best for inference)

**Performance difference between quantizations and parameter size**

Some benchmarks might be outdated due to the fast evolution of the LLM field.

* models requirements <https://www.reddit.com/r/LocalLLaMA/wiki/models/>
* **TheBloke :** each model published contains a table
* **Exllama** : [turboderp/exllama: A more memory-efficient rewrite of the HF transformers implementation of Llama for use with quantized weights. (github.com)](https://github.com/turboderp/exllama#new-implementation)
* **Reddit quantization benchmark**

<https://www.reddit.com/r/LocalLLaMA/comments/13l0j7m/a_comparative_look_at_ggml_quantization_and/>

* **memory requirements** <https://github.com/ggerganov/llama.cpp/blob/dc271c52ed65e7c8dfcbaaf84dabb1f788e4f3d0/README.md#memorydisk-requirements>
* llamma.cpp benchmarks <https://github.com/ggerganov/llama.cpp/blob/dc271c52ed65e7c8dfcbaaf84dabb1f788e4f3d0/README.md#quantization>

### What does 0 and 1 signify?

It's the naming convention GGML uses. They seem to name the quantization variations such that higher n usually is higher quality but uses more memory and has slower generation.

### How to download models ?

* Always read the how to download / how to use sections of the model.
* If you have a CPU you can download a GGML model via webui or manually from hugging face. You need a GGML model (pytorch models works but are too heavy and GPTQ models require GPU).
* A 4 bits or 5 bits model can work on a 16gb ram laptop perfectly since they need 3.9GB RAM . This model for example : [TheBloke/Llama-2-7b-Chat-GGUF · Hugging Face](https://huggingface.co/TheBloke/Llama-2-7b-Chat-GGUF)
* Manually : Download one GGUF/GGML file from hugging face, create a folder with its name , place the bin inside the created folder, then place the folder inside the folder models inside text generation webui.
* Download a model using the web-ui or manually from hugging face.
* Using GGML models works perfectly (do not put or rename the the folder a name that contains '5\\_bits' or GPTQ in the name of the model directory) GPTQ naming is for GPTQ models, and '5\\_bits' is for a specific file inside the GGML/GGUF model folders

Load the model (look in the terminal to get information about the model)

Generate text (look in the terminal for the number of tokens/s)

**Understand all parameters** <https://github.com/oobabooga/text-generation-webui#basic-settings>

**LLaMA.cpp parameters** <https://github.com/oobabooga/text-generation-webui#llamacpp>

**Advanced Docs** : [text-generation-webui/docs at main · oobabooga/text-generation-webui (github.com)](https://github.com/oobabooga/text-generation-webui/tree/main/docs)

Langforge : <https://github.com/ianarawjo/ChainForge>

Software built on LLM calls require one to verify the quality of outputs. ChainForge provides a suite of tools to evaluate and visualize prompt (and model) quality, with minimal effort by you. In other words, it aims to make evaluation of LLMs a piece of cake 🍰.