Local LLM Setup Guide (Mac) Ollama + Open Web-UI

Author: Winston Liang

You may find me on https://www.linkedin.com/in/winston-liang/ if you have any questions.

Section 1: Setting up a local LLM using Ollama and Open WebUI

In this guide, we will dive into the essentials of hosting a private LLM with our machine.

The main two Open-Source software to be used are: Ollama and Open Web-UI.

- Ollama: used to run the LLM in the backend.
- Open Web-UI: used as the front-end + backend, provides user account features, RAG, image as input etc…

There are other options to be considered instead of Ollama. For example, vLLM has been considered as faster than Ollama in processing multiple concurrent requests. But for simplicity, Ollama is the most straightforward one to implement.

1. **Install Ollama: https://ollama.com/**
2. **Download a model for Ollama**: https://ollama.com/search (e.g. deepseek-r1:1.5b), using command line interace.
    a. Run:

    ```
    ollama run deepseek-r1:1.5b # This opens up a chat dialog box in terminal

    # Basic commands:
    # To download a model: ollama pull (model_name)
    # To show all downloaded models: ollama list
    # To show all RUNNING models: ollama ps
    # To stop a model from running (to free up VRAM): ollama stop (model_name)
    ```

    b. If this is your first time running the model, it will automatically run “ollama pull deepseek-r1” to download the model.
    c. There are two types of models in general
        i. Reasoning models: they take time to think before responding to you. E.g. deepseek-r1:1.5b
        ii. Chat models: they quickly process your prompt and chat with you while thinking, like ChatGPT. E.g. qwen2.5:3b

3. **Install Open Web-UI:** https://docs.openwebui.com/getting-started/quick-start

    (If you are installing with Windows, you may refer to “Getting Started” from Open Web UI)

    1. Use Docker (Install: https://www.docker.com/products/docker-desktop/)
        a. Docker is used to automate the deployment of applications in lightweight containers so that applications can work efficiently in different environments and OS.

b. Using docker will make installation of Open Web-UI much easier.

4. **Setup:**
   a. **Setup an instance of the Open Web-UI**: https://docs.openwebui.com/getting-started/quick-start
   b. **Run Open Web-UI:** (go to localhost:8080 after this)

```
#If installed using docker, simply start the app in Docker Desktop

#If you installed Open Web-UI using uv (which is what I did in Mac, unrecommended):
DATA_DIR=~/.open-webui uvx --python 3.11 open-webui@latest serve


#If you installed using pip, you may refer to the document in Quick Start
```

   c. Create an admin account.
   d. Go to Admin Panel from the top right options. Find Connection and connect to Ollama.
   - If you have Ollama installed, it should be able to automatically detect it and call its API.

   e. Go to web search, enable it with engines like “duckduckgo” which does not require additional keys.
   - Highly recommended: SearXNG.
     - The highly commended by the community. A search engine that runs locally and privacy focused, removes metadata from our search requests etc.
     - Install here using docker: https://docs.searxng.org/admin/installation-docker.html
   - Google also provides API keys for searching, but is rate limited per day.

5. To **enable access from users in the network**, perform this in the terminal (https://github.com/ollama/ollama/issues/703#issuecomment-1951444576)

```
OLLAMA_HOST="0.0.0.0" ollama serve
```

6. Try allowing users to access the LLM server.
   - Find your ip address using “ifconfig” in terminal for Mac and Linux, ipconfig for Windows.
   - In the same network, the device can connect to the server using “YOUR_IP:8080”, assumming the default 8080 port is used.

Congratulations! You have set up the basics. You may explore:

- Retrieval-Augmented Generation: allows you to input custom documents as the knowledge base, so you do not have to upload it every time you ask the LLM.
  - https://docs.openwebui.com/tutorials/tips/rag-tutorial/
- Encryption options: quite important for your final production! https://docs.openwebui.com/getting-started/advanced-topics/https-encryption

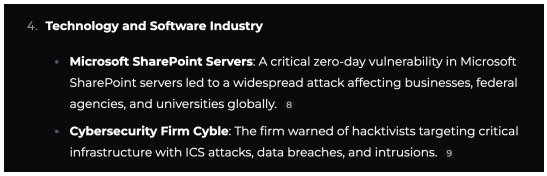
- More can be found on the official documentation: https://docs.openwebui.com/features/

Section 2: More to be explored and model consideration

1. Perplexica

Perplexica is an interface like Open WebUI which focuses on doing research and searching information from the Internet. It would be useful for things like security newsletter generation.

1. Link to repo https://github.com/ItzCrazyKns/Perplexica
    a. would recommend installing with docker!
    b. There are some online pipelines that allows you to call Perplexica API to be used in Open WebUI. You may look into this: https://openwebui.com/f/gwaanl/perplexica_pipe



2. Understanding models

- There are a few parameters to consider when choosing a model.
- Things you must know:
    - If the model size is larger than VRAM, then it cannot fully load into the VRAM, which will causing the performance to drop significantly.
        - How to reduce model size?
            1. Use smaller models
                a. models with a smaller number of parameters. E.g. qwen2.5:3b has 3 billion parameters.
            2. Use quantized models.
                a. they are models that have their bit representation reduced from the original full bit size models.
                b. https://www.reddit.com/r/LocalLLaMA/comments/1d32q63/what_do_the_numbersletters_in_the_quantized_model/
                c. Reduction from FP16 to Q4 (16 bit to 4 bit) only sees an ‘intelligence drop’ by around 15%, while massively reducing the model size (basically 0.25x of original size)
    - Benchmark:

        - Below are benchmarks on NVIDIA and Apple silicon GPUs on their inference speed. (LLama 7B model, on FP16, Q8 and Q4 Quantized models)
        - https://github.com/ggml-org/llama.cpp/discussions/15013
        - https://github.com/ggml-org/llama.cpp/discussions/4167
    - LLM Hallucinations:
        - Sometimes AI generates coherent but factually false information. It relates to how much training data it is fed on as well, meaning a smaller model is much more likely to be a risk on generating false information. You should try to choose a model (Open Sourced one) that has a lower hallucination rate.
        - Hallucination benchmark: https://github.com/vectara/hallucination-leaderboard
    - vLLM
        - If the production machine is not on Mac, but on Linux / Windows, you may consider replacing Ollama with vLLM if you wish to serve more concurrent users at a time.
        - https://naman1011.medium.com/ollama-vs-vllm-which-tool-handles-ai-models-better-a93345b911e6
    - LLM Performance Prediction
        - Here is a website that is very useful for calculating VRAM needed. However, inference speed predictions may not be reflective of the real world, as I have observed.
            - https://apxml.com/tools/vram-calculator