How to run AI on a local Raspberry Pi with Ollama (LLM) and Open WebUI

This article will demonstrate how to run local AI on a Raspberry Pi using Ollama and Open WebUI, based on practical testing rather than theory.

This article will demonstrate how to run AI locally on a Raspberry Pi using Ollama and Open WebUI, based on real-world testing rather than theory. This guide explains model selection, local setup, and practical solutions to common problems like storage limitations, memory errors, and overheating. It shows what actually works on a Raspberry Pi when deploying lightweight large language models (LLMs) for offline learning and demonstrations.

Recommended hardware and system configurations

Based on experience, this is what the article specifically recommends:

1. Raspberry Pi 4 (4GB or 8GB) or Raspberry Pi 5
2. An SD memory card of at least 32GB is required (16GB cards have caused numerous serious problems).
3. Suitable power source
4. Heat sink and fan
5. Raspberry Pi OS 64-bit operating system

Step-by-step guide on how to run AI on Raspberry Pi

Step 1: Update Raspberry Pi OS

Before installing anything AI-related, make sure your Raspberry Pi OS is fully updated. This step is easy to overlook, but in experience, outdated software packages can cause unexpected problems later when installing Ollama or Docker.

```
sudo apt update && sudo apt upgrade -y sudo reboot
```

Step 2: Install Ollama

Ollama is the core component that actually runs the AI ??models. It eliminates most of the complexity associated with managing language models, which is especially important on resource-constrained hardware.

```
curl-fsSL https://ollama.com/install.sh | sh
```

After installation, check using the command:

```
ollama --version
```

Step 3: Download and run the model

Downloading a simple model means downloading it to your machine so it can run locally. On a Raspberry Pi, model selection is crucial. Larger models might download successfully but fail during execution.

For the demo, TinyLlama was chosen because it loads stably and fits within the Raspberry Pi's memory limits.

```
ollama run tinyllama
```

Once loaded, test it with structured prompts such as summaries, rewrites, and checklists. These work well consistently. Avoid open-ended questions like 'Who are you?' as TinyLlama wasn't designed for that level of interaction.

Exit the session by pressing **Ctrl +** D.

Step 4: Install Docker

Install Docker to run Open WebUI in a container. This approach provides a clean setup and makes it easy to restart or remove the WebUI without affecting the rest of the system.

```
sudo apt install docker.io -y sudo systemctl enable docker sudo systemctl start
```

Confirm that Docker is working using the command:

```
docker --version
```

Step 5: Install Open WebUI

Open WebUI provides a complete chat interface in the browser while still utilizing local AI models. This makes the entire setup much more professional for demos.

```
sudo docker run -d  -p3000:8080  -v ollama:/root/.ollama  -v open-webui:/app/bac
ghcr.io/open-webui/open-webui:ollama
```

Next, access the user interface at:

1. http://localhost:3000 on Pi

2. http://ip:3000 from another device on the same network

Troubleshooting: Common Practical Problems

SD card is full (16GB card is faulty)

This is one of the biggest mistakes people often make initially. With a 16GB SD card, the system quickly runs out of space after loading models and installing Docker. When this happens, the Raspberry Pi becomes unresponsive and commands start to freeze.

Check disk usage using the command:

```
df -h
```

Cleaning up the system provided temporary relief, but the real solution is to upgrade to a 16GB or 64GB SD card.

Memory error and 'signal: killed' error

While running the models, many people repeatedly encountered errors where Ollama was shut down. This happens when Linux runs out of RAM and triggers the OOM killer.

Confirm this with the command:

```
dmesg -T | tail -n80 | egrep -i"oom|killed" free -h
```

Adding virtual memory (swap memory) has significantly improved stability:

```
sudo fallocate -l 4G /swapfile sudochmod600 /swapfile sudo mkswap /swapfile sudo
```

Overheating when loading the model.

During model loading and execution, the Raspberry Pi heats up noticeably. Without a proper cooling system, performance will decrease due to overheating.

Monitor the temperature using:

```
vcgencmd measure_temp
```

Adding a heatsink and fans made a noticeable difference.

Model management

Whenever you need to free up space or switch between demos, delete unused models:

```
ollama list ollama rm tinyllama
```

Running AI on a Raspberry Pi is entirely possible if you respect the hardware limitations. From experience, the key lessons are simple:

1. Storage capacity is more important than you think.
2. Virtual memory (swap) is essential on systems with low RAM.
3. A cooling system is required.
4. Small models require structured prompts.

This setup isn't intended to replace cloud-based AI, but it's perfect for learning, demonstrating, and understanding AI in real-world conditions.

You finished reading the article "**How to run AI on a local Raspberry Pi with Ollama (LLM) and Open WebUI**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.

© 2019 TipsMake.com