

Deploy and run LLM on Raspberry Pi 5 vs Raspberry Pi 4B (LLaMA, LLaMA2, Phi-2, Mixtral-MOE, etc)

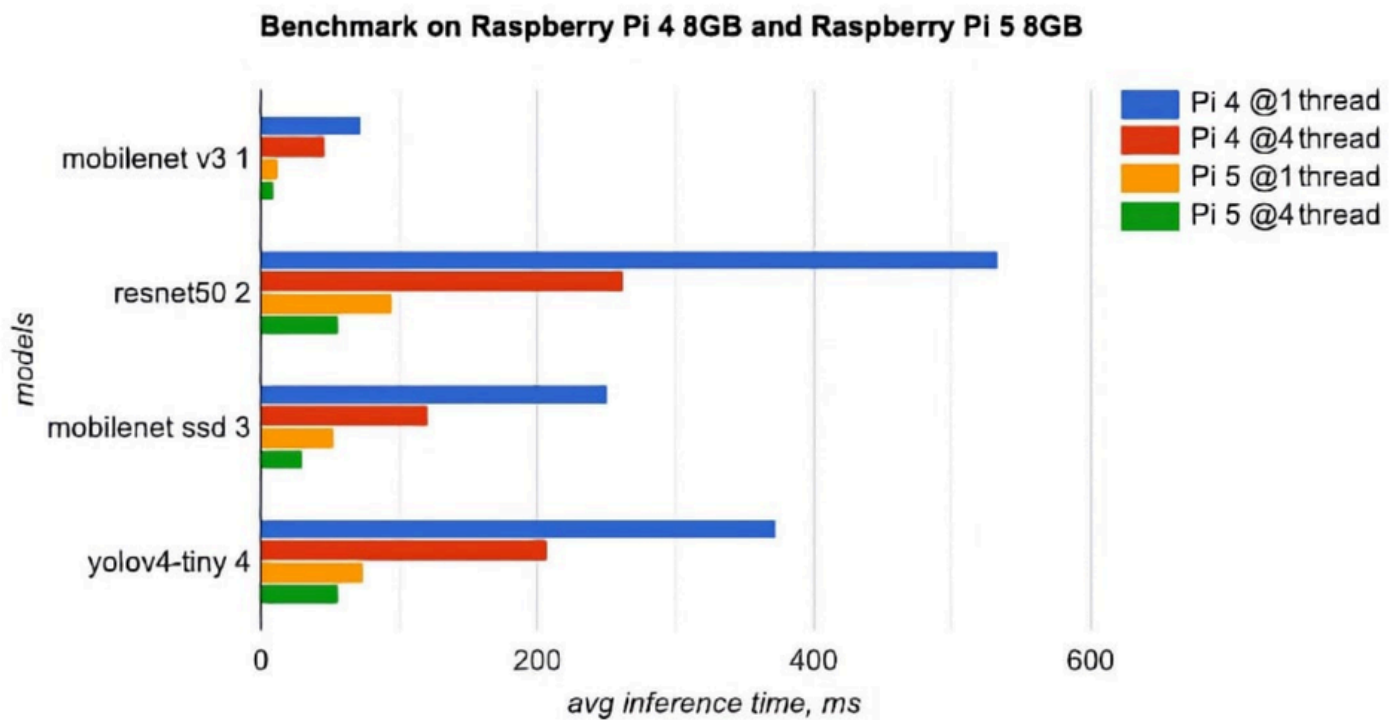
DFRobot Jan 02 2024  303867

This article will introduce how to deploy and run the recently popular LLM (large language models), including **LLaMA, LLaMA2, Phi-2, Mixtral-MOE, and mamba-gpt**, on the [Raspberry Pi 5 8GB](#). Compared to the [Raspberry Pi 4 model B](#), the Raspberry Pi 5 has upgrades in terms of processor, memory, and other aspects, resulting in some differences in performance and effectiveness. We will compare **the differences in running speed, resource usage, and model performance** among these LLMs to help you choose the right device for your needs and provide a reference for researching AI with limited hardware resources. At the same time, we will also discuss the key steps and matters, so that you can experience and test the running performance of LLMs on Raspberry Pi 5.

Specifications of Raspberry Pi 5 vs Raspberry Pi 4B

	Raspberry Pi 5	Raspberry Pi 4B
SoC	Broadcom BCM2712	Broadcom BCM2711
CPU	Quad-Core Cortex-A76 (ARM v8) 64-bit @ 2.4 GHz	Quad core Cortex-A72 (ARM v8) 64-bit @ 1.8 GHz
GPU	VideoCore VII @ 800 MHz Supports: OpenGL ES 3.1, Vulkan 1.2	VideoCore VI @ 500 MHz Supports: OpenGL ES 3.1, Vulkan 1.0
Display Output	2 x 4kp60 MINI HDMI Display Output Both can use 4kp60	2 x 4kp60 Mini HDMI One at 4Kp30 when both in use
Memory	LPDDR4X-4267 SDRAM 4GB, or 8GB	LPDDR4-3200 SDRAM 1GB, 2GB, 4GB or 8GB
Storage	Micro SD (SDR104 Compatible) M.2 NVME SSD Support via HAT	Micro SD
GPIO	40 PIN – Compatible with old Raspberry Pi HAT's	40 PIN
USB	2 x USB 2.0 2x USB 3.0 @ 5 Gbps	2 x USB 2.0 2 x USB 3.0
Connectors	2 x 4-lane MIPI camera / display transceivers PCIe 2.0 x1 Interface UART Breakout RTC Clock Power 4-Pin FAN Power	2-lane MIPI DSI Display Port 2-lane MIPI CSI Camera Port 4-Pole Stereo Audio and Composite Video Port
Networking	Dua-Band 802.11ac Bluetooth 5 / BLE Gigabit Ethernet PoE via POE + Hat (Incompatible with old version)	Dua-Band 802.11ac Bluetooth 5 / BLE Gigabit Ethernet PoE via POE + Hat
Power Button	Soft power button	None
Power Input	5V 4A via USB-C Port PoE via POE+ HAT (Incompatible with old version) 5V via GPIO	5V 3A via USB-C Port POE via POE+ HAT 5V via GPIO

Benchmarks on Raspberry Pi 5 8GB and Raspberry Pi 4 8GB



(From Alasdair Allan)

How to Choose LLM

LLM usually puts forward the prerequisite requirements for CPU/GPU in the project requirements. Since GPU inference LLM is temporarily unavailable on Raspberry Pi 5, we need to give priority to models that support CPU operation. In terms of model selection, due to the RAM limitation of Raspberry Pi 5, we need to give priority to models with smaller memory. Under normal circumstances, the model requires double the size of RAM to run normally. The quantized model has lower memory requirements, so we recommend using an 8GB Raspberry Pi 5 and a quantized small-scale model to experience and test LLM. Running effect on Raspberry Pi. The following list is a selection of smaller models from the open_llm_leaderboard on the Huggingface website, as well as the latest popular models.

Model	Average	ARC	HellaSwag	MMLU	TruthfulQA	Licens
mixtral_7bx4_moe	68.83	65.27	85.28	62.84	59.85	Mistra
Phi-2	61.33	61.09	75.11	58.11	44.47	Non-commr
mamba-gpt-7b-v1	58.61	61.26	84.1	63.46	46.34	Apach Licens
LLaMA2-7B-chat-hf	56.4	52.9	78.6	48.3	45.6	meta

LLaMA-13B	56.1	56.2	80.9	47.7	39.5	Non-commr
LLaMA-7B	49.7	51	77.8	35.7	34.3	Non-commr
ChatGLM-6B	48.2	38.8	59	46.7	48.1	Non-commr
Alpaca-7b	31.9	28.1	25.8	25.3	48.5	Non-commr

How to run LLM

After testing, since the GPU cannot be used to infer LLM on Raspberry Pi 5, we temporarily **use LLaMA.cpp and the CPU of Raspberry Pi 5** to infer each LLM. The following uses Phi-2 as an example to guide you in detail on how to deploy and run LLM on a Raspberry Pi 5 with 8GB RAM. At the same time, we will also discuss the key steps and matters needing attention, so that you can more quickly experience and test the running performance of LLM on Raspberry Pi 5.

PS: If you want to experience **Mixtral_moe**, please refer to:
<https://github.com/ggerganov/llama.cpp/tree/mixtral>

Environment Deployment

1. Deploying a virtual Python environment on the Raspberry Pi 5

```
sudo apt update && sudo apt install git
```

```
mkdir my_project
```

```
cd my_project
```

```
python -m venv env
```

```
source env/bin/activate
```

2. Download dependencies

```
python3 -m pip install torch numpy sentencepiece
```

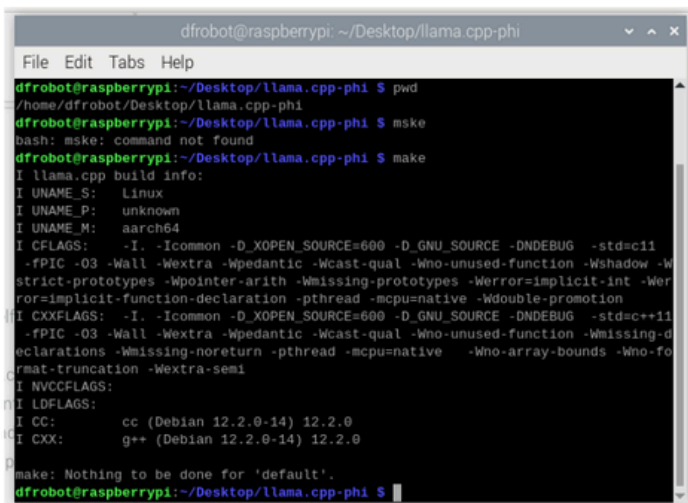
`sudo apt install g++ build-essential`

3. Download: <https://github.com/ggerganov/llama.cpp/tree/gg/phi-2>

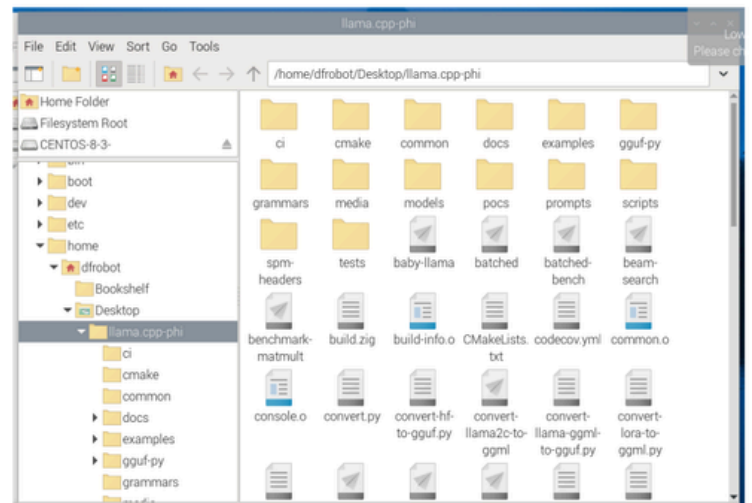
4. Build

`cd /home/dfrobot/Desktop/llama.cpp-phi`

`make`



```
dfrobot@raspberrypi: ~/Desktop/llama.cpp-phi
File Edit Tabs Help
dfrobot@raspberrypi:~/Desktop/llama.cpp-phi $ pwd
/home/dfrobot/Desktop/llama.cpp-phi
dfrobot@raspberrypi:~/Desktop/llama.cpp-phi $ mske
bash: mske: command not found
dfrobot@raspberrypi:~/Desktop/llama.cpp-phi $ make
I llama.cpp build info:
I UNAME_S: Linux
I UNAME_P: unknown
I UNAME_M: aarch64
I CFLAGS: -I. -Icommon -D_XOPEN_SOURCE=600 -D_GNU_SOURCE -DNDEBUG -std=c11 -fPIC -O3 -Wall -Wextra -Wpedantic -Wcast-qual -Wno-unused-function -Wshadow -Wstrict-prototypes -Wpointer-arith -Wmissing-prototypes -Werror=implicit-int -Werror=implicit-function-declaration -pthread -mcpu=native -Wdouble-promotion
I CXXFLAGS: -I. -Icommon -D_XOPEN_SOURCE=600 -D_GNU_SOURCE -DNDEBUG -std=c++11 -fPIC -O3 -Wall -Wextra -Wpedantic -Wcast-qual -Wno-unused-function -Wmissing-declarations -Wmissing-noreturn -pthread -mcpu=native -Wno-array-bounds -Wno-format-truncation -Wextra-semi
I NVCCFLAGS:
I LDFLAGS:
I CC: cc (Debian 12.2.0-14) 12.2.0
I CXX: g++ (Debian 12.2.0-14) 12.2.0
make: Nothing to be done for 'default'.
dfrobot@raspberrypi:~/Desktop/llama.cpp-phi $
```



Quantization

Model build quantization aims to reduce hardware requirements by reducing the accuracy of the weight parameters of each neuron in a deep neural network model. GGUF is a commonly used quantification method, allowing you to run LLMs on CPU or CPU + GPU. In general, the lower the number of bits and the more quantization, the smaller and faster the model will be, but at the expense of accuracy. For example, Q4 is the quantization method of the GGUF model file, which means using a 4-bit integer to quantize the weight of the model.

The 8GB RAM of the Raspberry Pi 5 is unsuitable for quantization models. We recommend quantizing the model on a Linux PC first and then copying the quantized files to the Raspberry Pi for deployment. On the Linux PC, after deploying the environment as in the previous step, use the `convert-hf-to-gguf.py` in LLaMA.cpp to convert the original Microsoft phi-2 model to GGUF format. The download URL for the original Microsoft phi-2 model:

<https://huggingface.co/microsoft/phi-2>

```
# convert hf model to GGUF
```

```
python convert-hf-to-gguf.py phi-2
```

```
# fp-16 inference
```

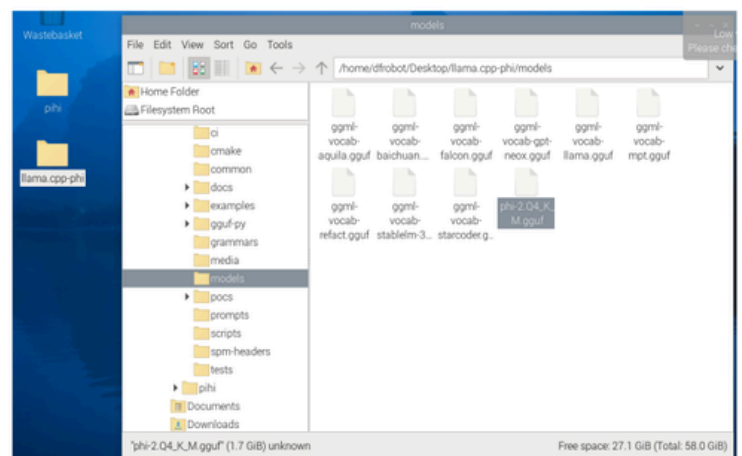
```
./main -m phi-2/ggml-model-f16.gguf -p "Question: Write a python function to print the first n numbers in the fibonacci series"
```

You can also directly search for already quantized GGUF model files on Huggingface and use the LLaMA.cpp to experience the model's performance quickly.

The original model is less than 6GB, while the Q4 quantized GGUF file is only 1.6GB in size. Q4-GGUF model URL: <https://huggingface.co/TheBloke/phi-2-GGUF/tree/main>

After downloading, place the model file in the directory: "llama.cpp-phi/models/".

README.md	26.1 kB	Upload	README.md
config.json	32 Bytes		GGUF model commit (made with ...)
phi-2.Q2_K.gguf	1.17 GB	LFS	GGUF model commit (made with ...)
phi-2.Q3_K_L.gguf	1.6 GB	LFS	GGUF model commit (made with ...)
phi-2.Q3_K_M.gguf	1.48 GB	LFS	GGUF model commit (made with ...)
phi-2.Q3_K_S.gguf	1.25 GB	LFS	GGUF model commit (made with ...)
phi-2.Q4_0.gguf	1.6 GB	LFS	GGUF model commit (made with ...)
phi-2.Q4_K_M.gguf	1.79 GB	LFS	GGUF model commit (made with ...)
phi-2.Q4_K_S.gguf	1.62 GB	LFS	GGUF model commit (made with ...)
phi-2.Q5_0.gguf	1.93 GB	LFS	GGUF model commit (made with ...)
phi-2.Q5_K_M.gguf	2.87 GB	LFS	GGUF model commit (made with ...)



Model Deployment

Running commands in the terminal of Raspberry Pi 5

```
./main -m models/phi-2.Q4_0.gguf -p "Question: Write a python function to print the first n numbers in the fibonacci series"
```

Summary

Test for Raspberry Pi 5 (8GB) & LLM

Model	File Size	Compatibility	Out of Memory	Token Speed
phi-2-Q4	1.7GB	✓		5.13 tokens/s
LLaMA-7B-Q4	< 4GB	✓		2.2 tokens/s
LLaMA2-7B-Q4	< 7GB	✓		2.3 tokens/s
LLaMA2-13B-Q4	< 4GB	✓		2.02 tokens/s
mixtral_7bx2_moe_Q4	< 8GB	✓		use llama.cpp <1 tokens/s
mamba-gpt-7b	<13GB		✓	

Test for Raspberry Pi 4B (8GB) & LLM

Model	File Size	Compatibility	Out of Memory	Token Speed
LLaMA-7B-Q4	< 4GB	✓		~0.1 tokens/s
Alpaca-7B-Q4	< 4GB	✓		
LLaMA2-7B-Q4	< 7GB	✓		~0.83 tokens/s
LLaMA-13B-Q4	< 8GB		✓	
ChatGLM-6B-Q4	13GB		✓	

Through analyzing the above table, it is not difficult to find that the running speed of LLM on Raspberry Pi 5 has significantly improved compared to Raspberry Pi 4B. [[Deploy and run LLM on Raspberry Pi 4B \(LLaMA, Alpaca, LLaMA2, ChatGLM\)](#)].

This indicates that Raspberry Pi 5 has stronger processing capabilities. As a resource-limited device, phi-2-Q4 performs particularly well, with an eval time speed of **5.13 tokens/s**, which undoubtedly demonstrates its excellent performance in processing speed.

In addition to the excellent performance of phi-2-Q4, LLaMA-7B-Q4, LLaMA2-7B-Q4, and LLaMA2-13B-Q4 also run satisfactorily on Raspberry Pi 5. However, it must be noted that for LLMs larger than 8GB, Raspberry Pi 5 still has limitations in loading the model, highlighting its RAM capacity constraints.

For LLM applications that require higher performance, [LattePanda Sigma](#) is a consideration. When running LLaMA2-7B-Q4, its speed can reach an astonishing **6 tokens/s**. [[Deploy and run LLM on LattePanda Sigma \(LLaMA, Alpaca, LLaMA2, ChatGLM\)](#)].

Overall, Raspberry Pi 5 has significantly improved in processing speed compared to its predecessor, but there are still limitations when dealing with large LLMs due to

[Store](#)[Community](#)[Forum](#)[Wiki](#)[Blog](#)[Learn](#)[\\$USD](#)

In summary, although Raspberry Pi 5 has significantly improved in processing speed compared to Raspberry Pi 4B, there are still some limitations when dealing with large LLMs. This provides new challenges and opportunities for future technological development.

More About AI Models

[1. Deploy and run LLM on Raspberry Pi 4B \(LLaMA, Alpaca, LLaMA2, ChatGLM\)](#)

[2. Deploy and run LLM on LattePanda Sigma \(LLaMA, Alpaca, LLaMA2, ChatGLM\)](#)

Related Product



Raspberry Pi 5 Single Board Computer -
8GB

\$80.00



Recent Blogs



embeddedworld
Exhibition & Conference
... it's a smarter world

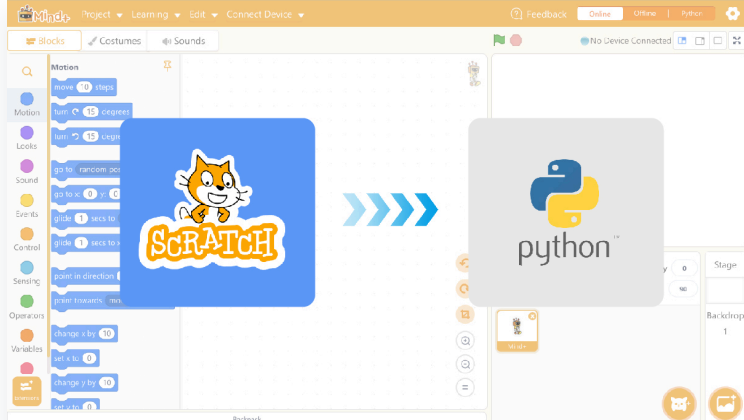
March 11-13
Hall 1-517

Connect with DFRobot at Embedded World: Embedde...

DFRobot is heading to Embedded World 2025 in Nuremberg, German...

NEWS

Mar 07 2025



Scratch to Python: Mindplus Bridges the Gap between...

This article introduces the Mind+ (Mindplus) editor, which seamlessl...

NEWS

Jan 23 2025



What is I3C: The Next Generation of I2C

I3C improves on I2C with faster speeds, better power efficiency, an...

NEWS

Jan 21 2025