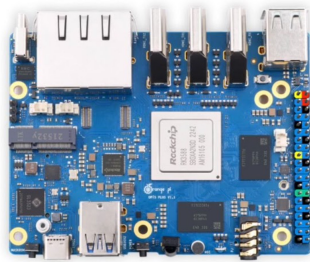


GPU-Accelerated LLM on a \$100 Orange Pi

Aug 9, 2023 • MLC Community

TL;DR

This post shows GPU-accelerated LLM running smoothly on an embedded device at a reasonable speed. More specifically, on a \$100 Orange Pi 5 with Mali GPU, we achieve 2.5 tok/sec for Llama2-7b and 5 tok/sec for RedPajama-3b through Machine Learning Compilation (MLC) techniques. Additionally, we are able to run a Llama-2 13b model at 1.5 tok/sec on a 16GB version of the Orange Pi 5+ under \$150.



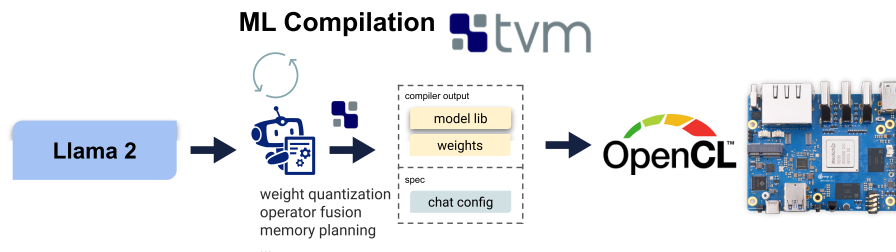
```
chris@chris-rk3588: ~/Documents/mlc_chat_cli
(GPT) chris@chris-rk3588:~/Documents/mlc_chat_cli$ mlc_chat_cli --local-id mlc-chat-llama-2-7b-chat-hf-q4f16_1
Use MLC config: "/home/chris/Documents/mlc_chat_cli/dist/prebuilt/mlc-chat-llama-2-7b-chat-hf-q4f16_1/mlc-chat-config.json"
Use model weights: "/home/chris/Documents/mlc_chat_cli/dist/prebuilt/mlc-chat-llama-2-7b-chat-hf-q4f16_1/weights-cache.json"
Use model library: "/home/chris/Documents/mlc_chat_cli/dist/prebuilt/lib/llama-2-7b-chat-hf-q4f16_1-openssl.so"
You can use the following special commands:
  /help      print the special commands
  /exit      quit the cli
  /stats     print out the latest stats (token/sec)
  /reset     restart a fresh chat
  /reload [local_id] reload model 'local_id' from disk, or reload the current model if 'local_id' is not specified
Loading model...
arm_release_ver: g13p0-01e0c0, rk_so_ver: 3
arm_release_ver of this libmali is 'g6p0-01e0c0', rk_so_ver is '7'.
Loading finished
Running system prompts...
System prompts finished
[INST]: write a three line poem about llama
[/INST]: Of course, I'd be happy to help! Here's a three-line poem about llamas:
Fluffy and gentle, with eyes so bright,
Llamas roam the Andes, with grace in sight,
Their woolly coats shine, in the sun's warm light.
[INST]: /stats
prefill: 4.9 tok/s, decode: 2.6 tok/s
[INST]:
```

Background

Progress in open language models has been catalyzing innovation across question-answering, translation, and creative tasks. While current solutions demand high-end desktop GPUs to achieve satisfactory performance, to unleash LLMs for everyday use, we wanted to understand how usable we could deploy them on the affordable embedded devices.

Many embedded devices come with mobile GPUs that can serve as a source of acceleration. In this post, we pick Orange Pi 5, a RK35888-based board that is similar to Raspberry Pi but also features a more powerful Mali-G610 GPU. This post summarizes our first attempt at leveraging Machine Learning Compilation and provides out-of-box GPU acceleration for this device.

Machine Learning Compilation for Mali



Machine learning compilation (MLC) is an emerging technology that automatically compiles and optimizes machine learning workloads, and deploys the compiled workload to a broad set of backends. At the time of writing, based on Apache TVM Unity, MLC supports platforms including browsers (WebGPU, WASM), NVIDIA GPUs (CUDA), AMD GPUs (ROCm, Vulkan), Intel GPUs (Vulkan), iOS and MacBooks (Metal), Android (OpenCL), and Mali GPUs (this post).

Generalizable ML Compilation for Mali Codegen

MLC is built on top of Apache TVM Unity, a generalizable stack for compiling machine learning models across different hardware and backends. To compile LLMs onto Mali GPUs, we reuse all the existing compilation pipeline without any code optimizations. More specifically, we successfully deployed Llama-2 and RedPajama models with the following steps:

- Reuse model optimization passes, including quantization, fusion, layout optimization, etc;
- Reuse a generic GPU kernel optimization space written in TVM TensorIR and re-target it to Mali GPUs;
- Reuse OpenCL codegen backend from TVM, and re-target it to Mali GPUs;
- Reuse the existing user interface, including Python APIs, CLI, and REST APIs.

Try it out

This section provides a step-by-step guide so that you can try it out on your own orange pi device. Here we use RedPajama-INCITE-Chat-3B-v1-q4f16_1 as the running example. You can replace that by Llama-2-7b-chat-hf-q4f16_1 or Llama-2-13b-chat-hf-q4f16_1 (requires a 16GB board).

Prepare

Please first follow the instruction [here](#), to setup the RK3588 board with OpenCL driver. Then clone the MLC-LLM from the source, and download weights and prebuilt libs.

```
# clone mlc-llm from GitHub
git clone --recursive https://github.com/mlc-ai/mlc-llm.git && cd mlc-llm
# Download prebuilt weights and libs
git lfs install
mkdir -p dist/prebuilt && cd dist/prebuilt
git clone https://github.com/mlc-ai/binary-mlc-llm-libs.git lib
git clone https://huggingface.co/mlc-ai/mlc-chat-RedPajama-INCITE-Chat-3B-v1-q4f16_1
cd ../../..
```

Try out the CLI

Build mlc_llm_cli from the source code

```
cd mlc-llm/
# create build directory
mkdir -p build && cd build
# generate build configuration
python3 ../cmake/gen_cmake_config.py
# build `mlc_chat_cli`
cmake .. && cmake --build . --parallel $(nproc)
&& cd ..
```

Verify installation

```
# expected to see `mlc_chat_cli`, `libmlc_llm.so`
and `libtvm_runtime.so`
ls -l ./build/
# expected to see help message
./build/mlc_chat_cli --help
```

Run LLMs through mlc_chat_cli

```
./build/mlc_chat_cli --local-id RedPajama-INCITE-Chat-3B-v1-q4f16_1 --device mali
```

```
You can use the following special commands:
/help      print the special commands
/exit     quit the cli
/stats    print out the latest stats (token/sec)
/reset    restart a fresh chat
/reload [local_id] reload model 'local_id' from disk, or reload the current model if 'local_id' is not specified

Loading model...
am_release_ver: g1p0@0eac0, rk_so_ver: 3
am_release_ver of this libmali is 'g6p0@0eac0', rk_so_ver is '7'.
Loading finished
Running system prompts...
System prompts finished
-humans: hello
-dot: hello
-humans: who are you
-dot: I am Open Assistant, a human-in-the-loop language model. As such, I am a language model trained by many people, so I don't have a name. I am able to generate human-like responses to your question s
-humans: write me a poem
-dot: In the end, all that remains is a memory
Of the first time, you kissed me so sweetly
The thrill that you gave me still lingers
On the back of my neck, a whisper in my ear
-humans: /stats
prefill: 6.1 tok/s, decode: 5.5 tok/s
-humans: write me a story about human and yosemite national park
-dot: It was a beautiful sunny day in Yosemite National Park. The granite monoliths stood proudly as if they were still watching over the valley. The air was crisp and the sun warmed your skin as you w
andered the meadows. You spotted a small band of spruce grouse in the meadow and marveled at their small bodies and long tails as you watched them run from the tall grasses. You gazed at the Half Dome,
rising high above the valley as if it were watching over you. You could have spent the entire day in the meadows, but you were a bit tired and decided to head back to your campsite. You picked up your b
ackpack and started your trek back to the campsite, wondering how your partner was doing.
-humans: /stats
prefill: 58.2 tok/s, decode: 5.0 tok/s
-humans: what is the meaning of the color red
-dot: Colors are used to convey information and meaning. Red is a color that is associated with fear and danger. It is also associated with love, passion, and anger. Red is a primary color which means th
at it has a lot of power and is often associated with power.
```

Try out the Python API

Build TVM runtime

```
# clone from GitHub
git clone --recursive https://github.com/mlc-ai/relax.git tvm_unity && cd tvm_unity/
# create build directory
mkdir -p build && cd build
# generate build configuration
cp ../cmake/config.cmake . && echo
"set(CMAKE_BUILD_TYPE
RelWithDebInfo)\nset(USE_OPENCL ON)" >>
config.cmake
# build `mlc_chat_cli`
cmake .. && cmake --build . --target runtime --
parallel $(nproc) && cd ../..
```

Setup python path (please set it to the bashrc or zshrc for persistent settings)

```
export TVM_HOME=$(pwd)/tvm_unity
export MLC_LLM_HOME=$(pwd)/mlc-llm
export
PYTHONPATH=$TVM_HOME/python:$MLC_LLM_HOME/python:${
```

Run the following python script.

```
from mlc_chat import ChatModule
from mlc_chat.callback import StreamToStdout
cm = ChatModule(model="RedPajama-INCITE-Chat-3B-
v1-q4f16_1")

# Generate a response for a given prompt
output = cm.generate(
    prompt="What is the meaning of life?",
    progress_callback=StreamToStdout(callback_interval=
)

# Print prefill and decode performance statistics
print(f"Statistics: {cm.stats()}\n")
```

Discussion and Future Work

Our current experiments show that 3B models might be a sweet spot. The RedPajama-3B model can provide up to 5 tok/sec and a decent chat experience. There is also room for improvements, specifically around the integer-to-float conversions. Moving forward, we will address the related issues and improve Mali GPUs' performance.

This post contributes to our quest to integrate LLMs into affordable devices and bring AI to everyone. Our future endeavors will focus on harnessing advancements in single-board computers, refining software frameworks like OpenCL and MLC-LLM, and exploring broader applications such as smart home devices. Collaborative efforts in the open-source community and a commitment to continuous learning and adaptation will be pivotal in navigating the evolving landscape of LLM deployment on emerging hardware.

Contributions

LLM on Orange Pi is primarily completed by [Haolin Zhang](#). The support of mali optimizations comes from Siyuan Feng, with foundation support from Junru Shao and Bohan Hou and other community members.