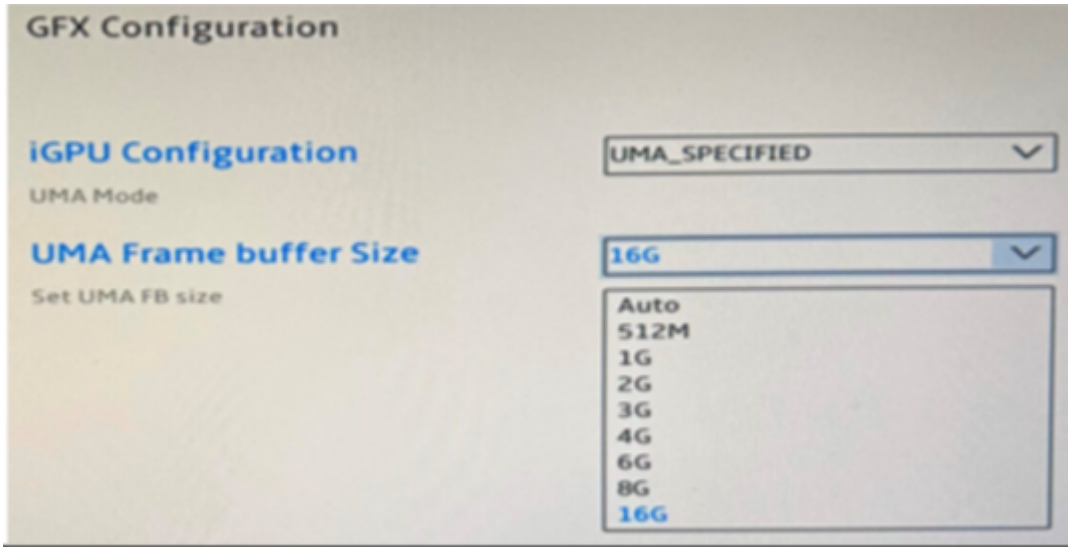
Run Ollama with AMD iGPU 780M-QuickStart

Test Platform

Platform	miniPC
HW	AMD Ryzen R8845HS + Radeon780M(iGPU, set 16GB VRAM of 64GB DDR)
OS	Ubuntu22.04
SW	ROCm6.0+PyTorch

Set UMA for iGPU in BOIS



Available CPU: Ryzen 7000/8000 with iGPU 780M

Prerequisites

1. Install GPU Driver and ROCm
   Refer to
   - AMD ROCm™ documentation — ROCm Documentation
   - Ubuntu native installation — ROCm installation (Linux) (amd.com)
     Steps:
     Install AMD GPU Driver
     `sudo apt install amdgpu-dkms`
     `sudo reboot`
     Install ROCm stack
     `sudo apt install rocm`

2. Install PyTorch-ROCm6.0
   I suggest to use conda to manage your environment
   `pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/rocm6.0`
3. Install Ollama (ollama/ollama: Get up and running with Llama 3, Mistral, Gemma, and other large language models. (github.com))
   `curl -fsSL https://ollama.com/install.sh | sh`

Benchmark

Quick test

`ollama run tinyllama "where was beethoven born?" --verbose`

`for run in {1..10}; do echo "where was beethoven born?" | ollama run tinyllama --verbose 2>&1 >/dev/null | grep "eval rate:"; done`

Model	Model Size	Radeon 780M (@ubuntu+ROCm6)	R8845HS (@Ubuntu)
tinyllama	637MB	92	72
llama2:latest	3.8GB	18	13
llama2-chinese	3.8GB	18	13
llama3:8b	4.7GB	16	12
qwen:1.8b	1.1GB	61	45

NOTE

- Performance in Tokens/s
- LLM is quantized as Q4_0 at default in Ollama

Steps

1. Stop the ollama.service
   `sudo systemctl stop ollama.service`
   Then find out the pid of ollama.service by 'ps -elf | grep ollama' and then 'kill -p [pid]'
2. for iGPU 780 w/ ROCm (not work in WSL, need run in Linux)
   *`HSA_OVERRIDE_GFX_VERSION="11.0.0" HCC_AMDGPU_TARGETS="gfx1103" OLLAMA_LLM_LIBRARY="rocm_v60002" ollama serve &
3. Run ollama
   `for run in {1..10}; do echo "Why is the sky blue?" | ollama run llama2:latest --verbose 2>&1 >/dev/null | grep "eval rate:"; done`

   NOTE Use rocm-smi to watch the utilization of iGPU When run ollama with ROCm

Another way to replace the step-2 above is to config the ollama.service be start with ROCm as defautl.
sudo systemctl edit ollama.service
Add the contents into the /etc/systemd/system/ollama.service.d/override.conf
[Service]
Environment="HSA_OVERRIDE_GFX_VERSION=11.0.0"
Environment="HCC_AMDGPU_TARGETS=gfx1103"
Environment="OLLAMA_LLM_LIBRARY=rocm_v60002"
Then Reboot the Linux or just restart the ollama.srevice by,
sudo system restart ollama.service

Examples of iGPU 780M w/ ROCm6
iGPU (780M)
HSA_OVERRIDE_GFX_VERSION="11.0.0" HCC_AMDGPU_TARGETS="gfx1103"
OLLAMA_LLM_LIBRARY="rocm_v60002" /usr/local/bin/ollama serve &
ollama run llama2:latest "where was beethoven born?" --verbose

Ludwig van Beethoven was born in Bonn, Germany on December 16, 1770.
total duration: 4.385911867s
load duration: 2.524807278s
prompt eval count: 27 token(s)
prompt eval duration: 465.157ms
prompt eval rate: 58.04 tokens/s
eval count: 26 token(s)
eval duration: 1.349772s
eval rate: 19.26 tokens/s

Check the log

```
$journalctl -u ollama.service > ollama_logs.txt
```

```
Apr 26 15:27:39 RyzerA ollama[1143]: ggml_cuda_init: GGML_CUDA_FORCE_MMQ:   yes
Apr 26 15:27:39 RyzerA ollama[1143]: ggml_cuda_init: CUDA_USE_TENSOR_CORES: no
Apr 26 15:27:39 RyzerA ollama[1143]: ggml_cuda_init: found 1 CUDA devices:
Apr 26 15:27:39 RyzerA ollama[1143]:   Device 0: NVIDIA GeForce RTX 4070 Laptop GPU, compute capability 8.9, VMM: yes
Apr 26 15:27:39 RyzerA ollama[1143]: llm_load_tensors: ggml ctx size =    0.22 MiB
Apr 26 15:27:39 RyzerA ollama[1143]: llm_load_tensors: offloading 32 repeating layers to GPU
Apr 26 15:27:39 RyzerA ollama[1143]: llm_load_tensors: offloading non-repeating layers to GPU
Apr 26 15:27:39 RyzerA ollama[1143]: llm_load_tensors: offloaded 33/33 layers to GPU
Apr 26 15:27:39 RyzerA ollama[1143]: llm_load_tensors:        CPU buffer size =   281.81 MiB
Apr 26 15:27:39 RyzerA ollama[1143]: llm_load_tensors:      CUDA0 buffer size =  4155.99 MiB
Apr 26 15:27:40 RyzerA ollama[1143]: ..................................................................................
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model: n_ctx      = 2048
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model: n_batch    = 512
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model: n_ubatch   = 512
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model: freq_base  = 500000.0
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model: freq_scale = 1
Apr 26 15:27:40 RyzerA ollama[1143]: llama_kv_cache_init:      CUDA0 KV buffer size =   256.00 MiB
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model: KV self size  =  256.00 MiB, K (f16):  128.00 MiB, V (f16):  128.00 MiB
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model:  CUDA_Host  output buffer size =     0.50 MiB
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model:      CUDA0 compute buffer size =   258.50 MiB
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model:  CUDA_Host compute buffer size =    12.01 MiB
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model: graph nodes  = 1030
Apr 26 15:27:40 RyzerA ollama[1143]: llama_new_context_with_model: graph splits = 2
```

Extension

Run LLM with PyTorch and HuggingFace at iGPU 780M

Step1: "pip insatll transformers"
Step2: set env for iGPU 780M
export HSA_OVERRIDE_GFX_VERSION=11.0.0
export HCC_AMDGPU_TARGETS=gfx1103
Step3: Run the LLM with scripts. (e.g. phi-2)

(rocm6) igpu@iHPT:~/phi-2$ python3 iphi-2.py

```
#examples-iphi-2.py
import torch
from transformers import AutoModelForCausalLM, AutoTokenizer
torch.set_default_device("cuda")

model = AutoModelForCausalLM.from_pretrained("microsoft/phi-2",
torch_dtype="auto", trust_remote_code=True)
tokenizer = AutoTokenizer.from_pretrained("microsoft/phi-2",
trust_remote_code=True)

inputs = tokenizer('''def print_prime(n):
"""
Print all primes between 1 and n
"""''', return_tensors="pt", return_attention_mask=False)

outputs = model.generate(**inputs, max_length=200)
text = tokenizer.batch_decode(outputs)[0]
print(text)
```