mistral_llama_large

May 2, 2024

[1]:
```
from llama_cpp import Llama
```

[2]:
```
#     --n_gpu_layers 43 --use_mlock true --n_ctx 200 --n_batch 200 --host 0.0.0.
 ↪0 --verbose true
llm = Llama(
      model_path="/data/DemoV1/Model4Demo/openhermes-2.5-mistral-7b.Q5_K_M.
 ↪gguf",
      n_gpu_layers=-1, # Uncomment to use GPU acceleration
      seed=1999, # Uncomment to set a specific seed
      n_ctx=200, # Uncomment to increase the context window
      n_batch=200,
      verbose=True,
)
```

```
llama_model_loader: loaded meta data with 20 key-value pairs and 291 tensors
from /data/DemoV1/Model4Demo/openhermes-2.5-mistral-7b.Q5_K_M.gguf (version GGUF
V3 (latest))
llama_model_loader: Dumping metadata keys/values. Note: KV overrides do not
apply in this output.
llama_model_loader: - kv   0:                       general.architecture str
= llama
llama_model_loader: - kv   1:                               general.name str
= teknium_openhermes-2.5-mistral-7b
llama_model_loader: - kv   2:                       llama.context_length u32
= 32768
llama_model_loader: - kv   3:                     llama.embedding_length u32
= 4096
llama_model_loader: - kv   4:                          llama.block_count u32
= 32
llama_model_loader: - kv   5:                  llama.feed_forward_length u32
= 14336
llama_model_loader: - kv   6:                 llama.rope.dimension_count u32
= 128
llama_model_loader: - kv   7:                 llama.attention.head_count u32
= 32
llama_model_loader: - kv   8:              llama.attention.head_count_kv u32
= 8
llama_model_loader: - kv   9:     llama.attention.layer_norm_rms_epsilon f32
```

```
= 0.000010
llama_model_loader: - kv  10:                      llama.rope.freq_base f32
= 10000.000000
llama_model_loader: - kv  11:                         general.file_type u32
= 17
llama_model_loader: - kv  12:                      tokenizer.ggml.model str
= llama
llama_model_loader: - kv  13:                     tokenizer.ggml.tokens
arr[str,32002]   = ["<unk>", "<s>", "</s>", "<0x00>", "<…
llama_model_loader: - kv  14:                     tokenizer.ggml.scores
arr[f32,32002]   = [0.000000, 0.000000, 0.000000, 0.0000…
llama_model_loader: - kv  15:                   tokenizer.ggml.token_type
arr[i32,32002]   = [2, 3, 3, 6, 6, 6, 6, 6, 6, 6, 6, 6, …
llama_model_loader: - kv  16:                tokenizer.ggml.bos_token_id u32
= 1
llama_model_loader: - kv  17:                tokenizer.ggml.eos_token_id u32
= 32000
llama_model_loader: - kv  18:            tokenizer.ggml.padding_token_id u32
= 0
llama_model_loader: - kv  19:               general.quantization_version u32
= 2
llama_model_loader: - type  f32:   65 tensors
llama_model_loader: - type q5_K:  193 tensors
llama_model_loader: - type q6_K:   33 tensors
llm_load_vocab: special tokens definition check successful ( 261/32002 ).
llm_load_print_meta: format           = GGUF V3 (latest)
llm_load_print_meta: arch             = llama
llm_load_print_meta: vocab type       = SPM
llm_load_print_meta: n_vocab          = 32002
llm_load_print_meta: n_merges         = 0
llm_load_print_meta: n_ctx_train      = 32768
llm_load_print_meta: n_embd           = 4096
llm_load_print_meta: n_head           = 32
llm_load_print_meta: n_head_kv        = 8
llm_load_print_meta: n_layer          = 32
llm_load_print_meta: n_rot            = 128
llm_load_print_meta: n_embd_head_k    = 128
llm_load_print_meta: n_embd_head_v    = 128
llm_load_print_meta: n_gqa            = 4
llm_load_print_meta: n_embd_k_gqa     = 1024
llm_load_print_meta: n_embd_v_gqa     = 1024
llm_load_print_meta: f_norm_eps       = 0.0e+00
llm_load_print_meta: f_norm_rms_eps   = 1.0e-05
llm_load_print_meta: f_clamp_kqv      = 0.0e+00
llm_load_print_meta: f_max_alibi_bias = 0.0e+00
llm_load_print_meta: f_logit_scale    = 0.0e+00
llm_load_print_meta: n_ff             = 14336
llm_load_print_meta: n_expert         = 0
```

```
llm_load_print_meta: n_expert_used    = 0
llm_load_print_meta: causal attn      = 1
llm_load_print_meta: pooling type     = 0
llm_load_print_meta: rope type        = 0
llm_load_print_meta: rope scaling     = linear
llm_load_print_meta: freq_base_train  = 10000.0
llm_load_print_meta: freq_scale_train = 1
llm_load_print_meta: n_yarn_orig_ctx  = 32768
llm_load_print_meta: rope_finetuned   = unknown
llm_load_print_meta: ssm_d_conv       = 0
llm_load_print_meta: ssm_d_inner      = 0
llm_load_print_meta: ssm_d_state      = 0
llm_load_print_meta: ssm_dt_rank      = 0
llm_load_print_meta: model type       = 8B
llm_load_print_meta: model ftype      = Q5_K - Medium
llm_load_print_meta: model params     = 7.24 B
llm_load_print_meta: model size       = 4.78 GiB (5.67 BPW)
llm_load_print_meta: general.name     = teknium_openhermes-2.5-mistral-7b
llm_load_print_meta: BOS token        = 1 '<s>'
llm_load_print_meta: EOS token        = 32000 '<|im_end|>'
llm_load_print_meta: UNK token        = 0 '<unk>'
llm_load_print_meta: PAD token        = 0 '<unk>'
llm_load_print_meta: LF token         = 13 '<0x0A>'
llm_load_print_meta: EOT token        = 32000 '<|im_end|>'
ggml_cuda_init: GGML_CUDA_FORCE_MMQ:   no
ggml_cuda_init: CUDA_USE_TENSOR_CORES: yes
ggml_cuda_init: found 2 CUDA devices:
  Device 0: Tesla V100-SXM2-16GB, compute capability 7.0, VMM: yes
  Device 1: Tesla V100-SXM2-16GB, compute capability 7.0, VMM: yes
llm_load_tensors: ggml ctx size =    0.44 MiB
llm_load_tensors: offloading 32 repeating layers to GPU
llm_load_tensors: offloading non-repeating layers to GPU
llm_load_tensors: offloaded 33/33 layers to GPU
llm_load_tensors:        CPU buffer size =    85.94 MiB
llm_load_tensors:      CUDA0 buffer size =  2352.25 MiB
llm_load_tensors:      CUDA1 buffer size =  2454.81 MiB
…
…
llama_new_context_with_model: n_ctx      = 256
llama_new_context_with_model: n_batch    = 200
llama_new_context_with_model: n_ubatch   = 200
llama_new_context_with_model: flash_attn = 0
llama_new_context_with_model: freq_base  = 10000.0
llama_new_context_with_model: freq_scale = 1
llama_kv_cache_init:      CUDA0 KV buffer size =    16.00 MiB
llama_kv_cache_init:      CUDA1 KV buffer size =    16.00 MiB
llama_new_context_with_model: KV self size  =   32.00 MiB, K (f16):   16.00 MiB,
V (f16):   16.00 MiB
```

```
llama_new_context_with_model:  CUDA_Host  output buffer size =     0.12 MiB
llama_new_context_with_model: pipeline parallelism enabled (n_copies=4)
llama_new_context_with_model:      CUDA0 compute buffer size =    44.63 MiB
llama_new_context_with_model:      CUDA1 compute buffer size =    44.63 MiB
llama_new_context_with_model:  CUDA_Host compute buffer size =     4.01 MiB
llama_new_context_with_model: graph nodes  = 1030
llama_new_context_with_model: graph splits = 3
AVX = 1 | AVX_VNNI = 0 | AVX2 = 1 | AVX512 = 1 | AVX512_VBMI = 0 | AVX512_VNNI =
0 | FMA = 1 | NEON = 0 | ARM_FMA = 0 | F16C = 1 | FP16_VA = 0 | WASM_SIMD = 0 |
BLAS = 1 | SSE3 = 1 | SSSE3 = 1 | VSX = 0 | MATMUL_INT8 = 0 | LLAMAFILE = 1 |
Model metadata: {'tokenizer.ggml.padding_token_id': '0',
'tokenizer.ggml.eos_token_id': '32000', 'general.architecture': 'llama',
'llama.rope.freq_base': '10000.000000', 'llama.context_length': '32768',
'general.name': 'teknium_openhermes-2.5-mistral-7b', 'llama.embedding_length':
'4096', 'llama.feed_forward_length': '14336',
'llama.attention.layer_norm_rms_epsilon': '0.000010',
'llama.rope.dimension_count': '128', 'tokenizer.ggml.bos_token_id': '1',
'llama.attention.head_count': '32', 'llama.block_count': '32',
'llama.attention.head_count_kv': '8', 'general.quantization_version': '2',
'tokenizer.ggml.model': 'llama', 'general.file_type': '17'}
Using fallback chat format: None
```

[3]:
```python
output = llm(
      "Q: what is the capital city of France? A: ", # Prompt
      max_tokens=None, # Generate up to 32 tokens, set to None to generate up␣
 ↪to the end of the context window
      stop=["Q:", "\n"], # Stop generating just before the model would generate␣
 ↪a new question
      echo=True # Echo the prompt back in the output
) # Generate a completion, can also call create_completion
print(output)
```

```
llama_print_timings:        load time =     138.75 ms
llama_print_timings:      sample time =       1.32 ms /     3 runs   (    0.44
ms per token,  2267.57 tokens per second)
llama_print_timings: prompt eval time =     138.66 ms /    14 tokens (    9.90
ms per token,   100.97 tokens per second)
llama_print_timings:        eval time =      25.95 ms /     2 runs   (   12.97
ms per token,    77.07 tokens per second)
llama_print_timings:       total time =     188.42 ms /    16 tokens
```

```
{'id': 'cmpl-c21cccde-8e0c-46a8-a1de-e9553c04adea', 'object': 'text_completion',
'created': 1714672825, 'model':
'/data/DemoV1/Model4Demo/openhermes-2.5-mistral-7b.Q5_K_M.gguf', 'choices':
[{'text': 'Q: what is the capital city of France? A: prepare AUTH', 'index': 0,
'logprobs': None, 'finish_reason': 'stop'}], 'usage': {'prompt_tokens': 14,
'completion_tokens': 2, 'total_tokens': 16}}
```

[4]:
```
#print(output['choices'][0]['text'])
```

[5]:
```
output = llm(
      "Q: what is the capital city of United Kingdom? A: ", # Prompt
      max_tokens=None, # Generate up to 32 tokens, set to None to generate up␣
 ↪to the end of the context window
      stop=["Q:", "\n"], # Stop generating just before the model would generate␣
 ↪a new question
      echo=True # Echo the prompt back in the output
) # Generate a completion, can also call create_completion
print(output)
```

```
Llama.generate: prefix-match hit

llama_print_timings:        load time =     138.75 ms
llama_print_timings:      sample time =       4.79 ms /     9 runs   (    0.53
ms per token,  1876.96 tokens per second)
llama_print_timings: prompt eval time =      27.47 ms /     6 tokens (    4.58
ms per token,   218.39 tokens per second)
llama_print_timings:        eval time =      99.12 ms /     8 runs   (   12.39
ms per token,    80.71 tokens per second)
llama_print_timings:       total time =     209.79 ms /    14 tokens
{'id': 'cmpl-b89062a7-5de0-41c4-93df-4862e3ec9e65', 'object': 'text_completion',
'created': 1714672834, 'model':
'/data/DemoV1/Model4Demo/openhermes-2.5-mistral-7b.Q5_K_M.gguf', 'choices':
[{'text': 'Q: what is the capital city of United Kingdom? A: isecondsęchéARKPref
acc Wars<|im_start|>', 'index': 0, 'logprobs': None, 'finish_reason': 'stop'}],
'usage': {'prompt_tokens': 15, 'completion_tokens': 8, 'total_tokens': 23}}
```

[ ]:
```

```

[6]:
```
output = llm(
      "Q: what is the capital city of Scotland? A: ", # Prompt
      max_tokens=None, # Generate up to 32 tokens, set to None to generate up␣
 ↪to the end of the context window
      stop=["Q:", "\n"], # Stop generating just before the model would generate␣
 ↪a new question
      echo=True # Echo the prompt back in the output
) # Generate a completion, can also call create_completion
print(output)
```

```
Llama.generate: prefix-match hit

llama_print_timings:        load time =     138.75 ms
llama_print_timings:      sample time =       1.96 ms /     4 runs   (    0.49
ms per token,  2038.74 tokens per second)
llama_print_timings: prompt eval time =      21.16 ms /     5 tokens (    4.23
```

```
ms per token,   236.24 tokens per second)
llama_print_timings:        eval time =      36.57 ms /     3 runs   (   12.19
ms per token,    82.03 tokens per second)
llama_print_timings:       total time =      88.34 ms /     8 tokens
```

```
{'id': 'cmpl-3f240e08-b948-4ba6-a7ef-d7bf2ff805ab', 'object': 'text_completion',
'created': 1714672843, 'model':
'/data/DemoV1/Model4Demo/openhermes-2.5-mistral-7b.Q5_K_M.gguf', 'choices':
[{'text': 'Q: what is the capital city of Scotland? A:  Rober<|im_start|>ouri',
'index': 0, 'logprobs': None, 'finish_reason': 'stop'}], 'usage':
{'prompt_tokens': 14, 'completion_tokens': 3, 'total_tokens': 17}}
```

[10]:
```
import llama_cpp as llcpp
llcpp.__version__
```

[10]:
```
'0.2.68'
```

[ ]:
```

```