Module_2_Maven_TextStreamer_Meta_Llama_3_1_8B_Instruct

August 15, 2025

[1]:
```
%%capture
#!pip install transformers
#!pip install bitsandbytes
```

[2]:
```
# from huggingface_hub import notebook_login
# notebook_login()
```

0.1 Using TextStreamer

[3]:
```
from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline,␣
↪TextStreamer
import torch
from transformers import BitsAndBytesConfig
```

[4]:
```
def start_gpu_stat():
    #@title Show current memory stats
    #Set torch device to get properties global: torch.cuda.set_device(0)
    gpu_stats = torch.cuda.get_device_properties(0)
    initial_gpu_memory = round(torch.cuda.max_memory_reserved() / 1024 / 1024 /␣
↪1024, 3)
    max_memory = round(gpu_stats.total_memory / 1024 / 1024 / 1024, 3)
    return initial_gpu_memory, max_memory

def final_gpu_stat(_initial_gpu_memory, _max_memory):
    #@title Show final memory and time stats
    used_memory = round(torch.cuda.max_memory_reserved() / 1024 / 1024 / 1024,␣
↪3)
    used_memory_for_diff = round(used_memory - _initial_gpu_memory, 3)
    used_percentage = round(used_memory         /_max_memory*100, 3)
    diff_percentage = round(used_memory_for_diff/_max_memory*100, 3)

    print(f"Max memory = {_max_memory} GB.")
    print(f"{_initial_gpu_memory} GB of INITIAL memory reserved.")
    print(f"Peak reserved FINAL memory = {used_memory} GB.")
    print(f"Peak reserved memory DIFFERENCE = {used_memory_for_diff} GB.")
    print(f"Peak reserved memory % of FINAL memory = {used_percentage} %.")
```

```
    print(f"Peak reserved memory % of DIFFERENCE memory = {diff_percentage} %.")
```

1 Text Streaming Without Quantization

[5]:
```
%env HF_HOME=/mnt/sda1/huggingface
model_id = "unsloth/Meta-Llama-3.1-8B-Instruct" # Replace with your model

# Load tokenizer and model in 4-bit
tokenizer = AutoTokenizer.from_pretrained(model_id)

initial_gpu_memory, max_memory = start_gpu_stat()

model = AutoModelForCausalLM.from_pretrained(
    model_id,
    device_map="auto"
)

final_gpu_stat(initial_gpu_memory, max_memory)
```

```
env: HF_HOME=/mnt/sda1/huggingface
```

```
Loading checkpoint shards:   0%|          | 0/4 [00:00<?, ?it/s]
```

```
Max memory = 47.413 GB.
0.0 GB of INITIAL memory reserved.
Peak reserved FINAL memory = 31.332 GB.
Peak reserved memory DIFFERENCE = 31.332 GB.
Peak reserved memory % of FINAL memory = 66.083 %.
Peak reserved memory % of DIFFERENCE memory = 66.083 %.
```

[6]:
```
from transformers import TextStreamer


# Define Alpaca-style prompt format
alpaca_prompt = """Below is an instruction that describes a task. Write a␣
 ↪response that appropriately completes the request.

### Instruction:
{}

### Response:
"""

# Prepare input text
prompt_text = alpaca_prompt.format("What is the importance of using renewable␣
 ↪energy?")  # instruction
```

```
inputs = tokenizer([prompt_text], return_tensors="pt").to(model.device)  # Move␣
 ↪inputs to model's device

# Initialize text streamer
text_streamer = TextStreamer(tokenizer, skip_prompt=True,␣
 ↪skip_special_tokens=False)

# Generate response with streamer
_ = model.generate(**inputs, streamer=text_streamer, max_new_tokens=100)
```

```
The importance of using renewable energy is multifaceted and crucial for the
future of our planet. Firstly, renewable energy is a sustainable and
environmentally friendly alternative to fossil fuels, which are finite resources
that contribute to climate change and pollution. By transitioning to renewable
energy sources such as solar, wind, and hydroelectric power, we can
significantly reduce our carbon footprint and mitigate the effects of global
warming. Additionally, renewable energy can help to improve air quality, reduce
greenhouse gas emissions, and protect ecosystems. Furthermore,
```

[7]:
```
from transformers import TextIteratorStreamer
from threading import Thread
import time

# Define Alpaca-style prompt format
alpaca_prompt = """Below is an instruction that describes a task. Write a␣
 ↪response that appropriately completes the request.

### Instruction:
{}

### Response:
"""

# Prepare input text
prompt_text = alpaca_prompt.format("What is the importance of using renewable␣
 ↪energy?")
inputs = tokenizer(prompt_text, return_tensors="pt").to(model.device)

# Initialize variables for time measurements
start_time = time.time()
token_times = []

# Initialize streamer
streamer = TextIteratorStreamer(tokenizer, skip_prompt=True,␣
 ↪skip_special_tokens=False)

# Start generation in a separate thread
```

```
thread = Thread(target=model.generate, kwargs={
    'input_ids': inputs['input_ids'],
    'attention_mask': inputs['attention_mask'],
    'streamer': streamer,
    'max_new_tokens': 100
})
thread.start()

# Initialize a variable to store the model output
model_output = ""
first_token_time = None

# Iterate over the streamer to get the generated text in chunks
for i, new_text in enumerate(streamer):
    model_output += new_text
    print(new_text, end='')

    # Measure time for the first token
    if i == 0:
        first_token_time = time.time()
    # Measure time for each token
    token_times.append(time.time())

# Calculate end-to-end latency
end_time = time.time()
end_to_end_latency = end_time - start_time

# Calculate time to first token
ttft = first_token_time - start_time if first_token_time else 0

# Calculate inter-token latency
itl = sum(x - y for x, y in zip(token_times[1:], token_times[:-1])) /␣
 ↪(len(token_times) - 1) if len(token_times) > 1 else 0

# Calculate throughput
throughput = len(tokenizer.encode(model_output)) / end_to_end_latency if␣
 ↪model_output else 0

print("\nTime To First Token (TTFT):", ttft)
print("Inter-token latency (ITL):", itl)
print("End-to-end Latency:", end_to_end_latency)
print("Throughput:", throughput)
```

```
Renewable energy is important because it is a sustainable and environmentally
friendly way to generate power. It reduces our reliance on fossil fuels, which
contribute to climate change, air pollution, and water pollution. Renewable
energy sources like solar, wind, and hydroelectric power are abundant and can be
```

```
replenished naturally, unlike fossil fuels, which are finite resources. Using
renewable energy can help mitigate the negative impacts of climate change,
improve air and water quality, and ensure a cleaner and healthier environment
for future generations.
Time To First Token (TTFT): 0.07187366485595703
Inter-token latency (ITL): 0.0466264820098877
End-to-end Latency: 4.734689950942993
Throughput: 21.331914242850083
```

2 Text Streaming With Quantization

Shutdown and Restart the kernel before starting below cells

[8]:
```
from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline,␣
↪TextStreamer
import torch
from transformers import BitsAndBytesConfig
```

[9]:
```
def start_gpu_stat():
    #@title Show current memory stats
    #Set torch device to get properties global: torch.cuda.set_device(0)
    gpu_stats = torch.cuda.get_device_properties(0)
    initial_gpu_memory = round(torch.cuda.max_memory_reserved() / 1024 / 1024 /␣
↪1024, 3)
    max_memory = round(gpu_stats.total_memory / 1024 / 1024 / 1024, 3)
    return initial_gpu_memory, max_memory

def final_gpu_stat(_initial_gpu_memory, _max_memory):
    #@title Show final memory and time stats
    used_memory = round(torch.cuda.max_memory_reserved() / 1024 / 1024 / 1024,␣
↪3)
    used_memory_for_diff = round(used_memory - _initial_gpu_memory, 3)
    used_percentage = round(used_memory         /_max_memory*100, 3)
    diff_percentage = round(used_memory_for_diff/_max_memory*100, 3)

    print(f"Max memory = {_max_memory} GB.")
    print(f"{_initial_gpu_memory} GB of INITIAL memory reserved.")
    print(f"Peak reserved FINAL memory = {used_memory} GB.")
    print(f"Peak reserved memory DIFFERENCE = {used_memory_for_diff} GB.")
    print(f"Peak reserved memory % of FINAL memory = {used_percentage} %.")
    print(f"Peak reserved memory % of DIFFERENCE memory = {diff_percentage} %.")
```

[10]:
```
model_id = "unsloth/Meta-Llama-3.1-8B-Instruct" # Replace with your model

# 4-bit quantization configuration
quantization_config = BitsAndBytesConfig(
    load_in_4bit=True,
```

```
    bnb_4bit_compute_dtype=torch.float16,
    bnb_4bit_quant_type="nf4"
)

# Load tokenizer and model in 4-bit
tokenizer = AutoTokenizer.from_pretrained(model_id)

initial_gpu_memory, max_memory = start_gpu_stat()

model = AutoModelForCausalLM.from_pretrained(
    model_id,
    quantization_config=quantization_config,
    device_map="auto"
)

final_gpu_stat(initial_gpu_memory, max_memory)
```

```
Loading checkpoint shards:   0%|          | 0/4 [00:00<?, ?it/s]
```

```
Max memory = 47.413 GB.
31.381 GB of INITIAL memory reserved.
Peak reserved FINAL memory = 37.436 GB.
Peak reserved memory DIFFERENCE = 6.055 GB.
Peak reserved memory % of FINAL memory = 78.957 %.
Peak reserved memory % of DIFFERENCE memory = 12.771 %.
```

[11]:
```
from transformers import TextStreamer

# Define Alpaca-style prompt format
alpaca_prompt = """Below is an instruction that describes a task. Write a␣
 ↪response that appropriately completes the request.

### Instruction:
{}

### Response:
"""

# Prepare input text
prompt_text = alpaca_prompt.format("What is the importance of using renewable␣
 ↪energy?")  # instruction

inputs = tokenizer([prompt_text], return_tensors="pt").to(model.device)  # Move␣
 ↪inputs to model's device

# Initialize text streamer
text_streamer = TextStreamer(tokenizer, skip_prompt=True,␣
 ↪skip_special_tokens=False)
```

```
# Generate response with streamer
_ = model.generate(**inputs, streamer=text_streamer, max_new_tokens=100)
```

```
The importance of using renewable energy lies in its ability to reduce our
reliance on finite fossil fuels, mitigate climate change, and promote energy
security. Renewable energy sources, such as solar and wind power, are
sustainable and can be replenished naturally, unlike fossil fuels which are
depleting. This shift towards renewable energy helps reduce greenhouse gas
emissions, which are a major contributor to global warming. Furthermore,
renewable energy can provide energy independence, improve air quality, and
create jobs in the renewable energy sector. Additionally
```

[13]:
```
from transformers import TextIteratorStreamer
from threading import Thread
import time

# Define Alpaca-style prompt format
alpaca_prompt = """Below is an instruction that describes a task. Write a␣
 ↪response that appropriately completes the request.

### Instruction:
{}

### Response:
"""

# Prepare input text
prompt_text = alpaca_prompt.format("What is the importance of using renewable␣
 ↪energy?")
inputs = tokenizer(prompt_text, return_tensors="pt").to(model.device)

# Initialize variables for time measurements
start_time = time.time()
token_times = []

# Initialize streamer
streamer = TextIteratorStreamer(tokenizer, skip_prompt=True,␣
 ↪skip_special_tokens=False)

# Start generation in a separate thread
thread = Thread(target=model.generate, kwargs={
    'input_ids': inputs['input_ids'],
    'attention_mask': inputs['attention_mask'],
    'streamer': streamer,
    'max_new_tokens': 100
})
```

```
thread.start()

# Initialize a variable to store the model output
model_output = ""
first_token_time = None

# Iterate over the streamer to get the generated text in chunks
for i, new_text in enumerate(streamer):
    model_output += new_text
    print(new_text, end='')

    # Measure time for the first token
    if i == 0:
        first_token_time = time.time()
    # Measure time for each token
    token_times.append(time.time())

# Calculate end-to-end latency
end_time = time.time()
end_to_end_latency = end_time - start_time

# Calculate time to first token
ttft = first_token_time - start_time if first_token_time else 0

# Calculate inter-token latency
itl = sum(x - y for x, y in zip(token_times[1:], token_times[:-1])) /␣
 ↪(len(token_times) - 1) if len(token_times) > 1 else 0

# Calculate throughput
throughput = len(tokenizer.encode(model_output)) / end_to_end_latency if␣
 ↪model_output else 0

print("\nTime To First Token (TTFT):", ttft)
print("Inter-token latency (ITL):", itl)
print("End-to-end Latency:", end_to_end_latency)
print("Throughput:", throughput)
```

```
The importance of using renewable energy is multifaceted and crucial for the
sustainability of our planet. Firstly, it helps reduce our reliance on fossil
fuels, which are finite resources that contribute to climate change and air
pollution. By transitioning to renewable energy sources such as solar, wind,
hydro, and geothermal power, we can decrease greenhouse gas emissions, mitigate
the effects of global warming, and improve air quality. This not only protects
the environment but also ensures a cleaner and healthier living space for future
generations.
```

```
Time To First Token (TTFT): 0.07175731658935547
Inter-token latency (ITL): 0.03085422992706299
End-to-end Latency: 3.1573333740234375
Throughput: 31.98901985801207
```

[ ]: