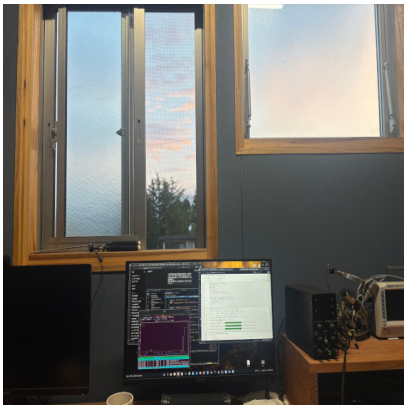


ç?¥è?è??ç©•ã?»å ±æ??ã•®ç ?ç©¶ã??ã••ã•®2 Unslothã??ä½¿ã•£ã• ?ã??ã?¡ã?¤ã?³ã?•ã?¥ã?¼ã??ã?³ã?°ã•®è©¦ã•¿

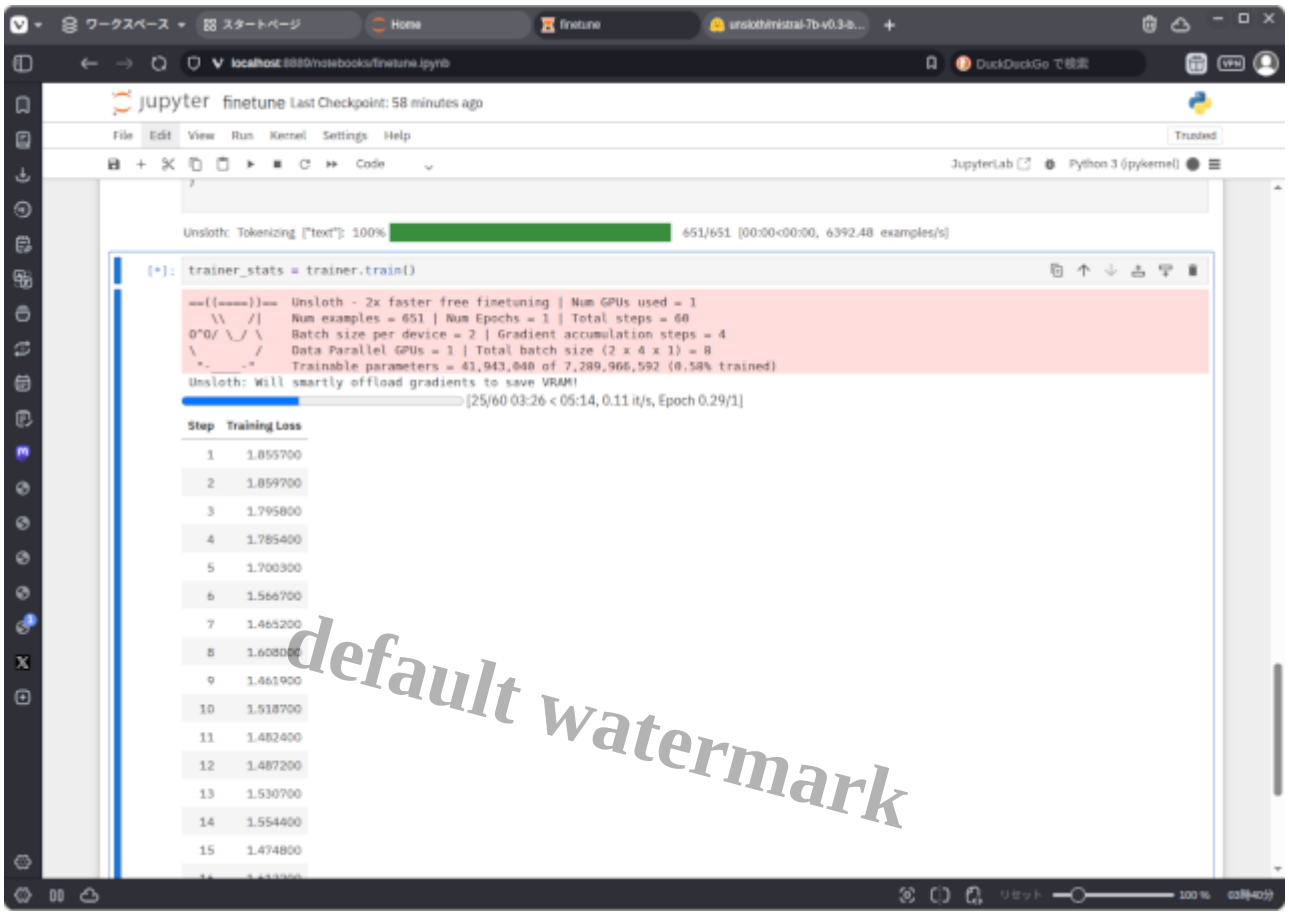
Description

default watermark

æ?¼é??ã•¯æ??ã•ã•¦ä½?ã??ã•?ã??æ°?ã•«ã•ªã??ã•¾ã•?ã??ã??æ??è¿?ã•¯ã?•å®?å ¨ã• «æ?¼å¤?é??è»¢ã•?ã•?æ?¥ã? ã??é?•ã•£ã•¦ã•?ã•¾ã•?ã??æ?¨å¤?ã•?ã??ä»?æ?•ã•«ã•?ã•?ã• ¦ã?³ã?³ã??ã?¥ã?¼ã?¿ã?¼ã•®å?•ã•«åº§ã•£ã•¦LLMã•®ã?•ã?¥ã?¼ã??ã?³ã?°ã??è©¦ã•¿ã•¦ã•?ã•¾ã•?ã• ?ã??æ?•æ?¥ã•?ç?©ã•?ã•?ã•§ã•?ã??

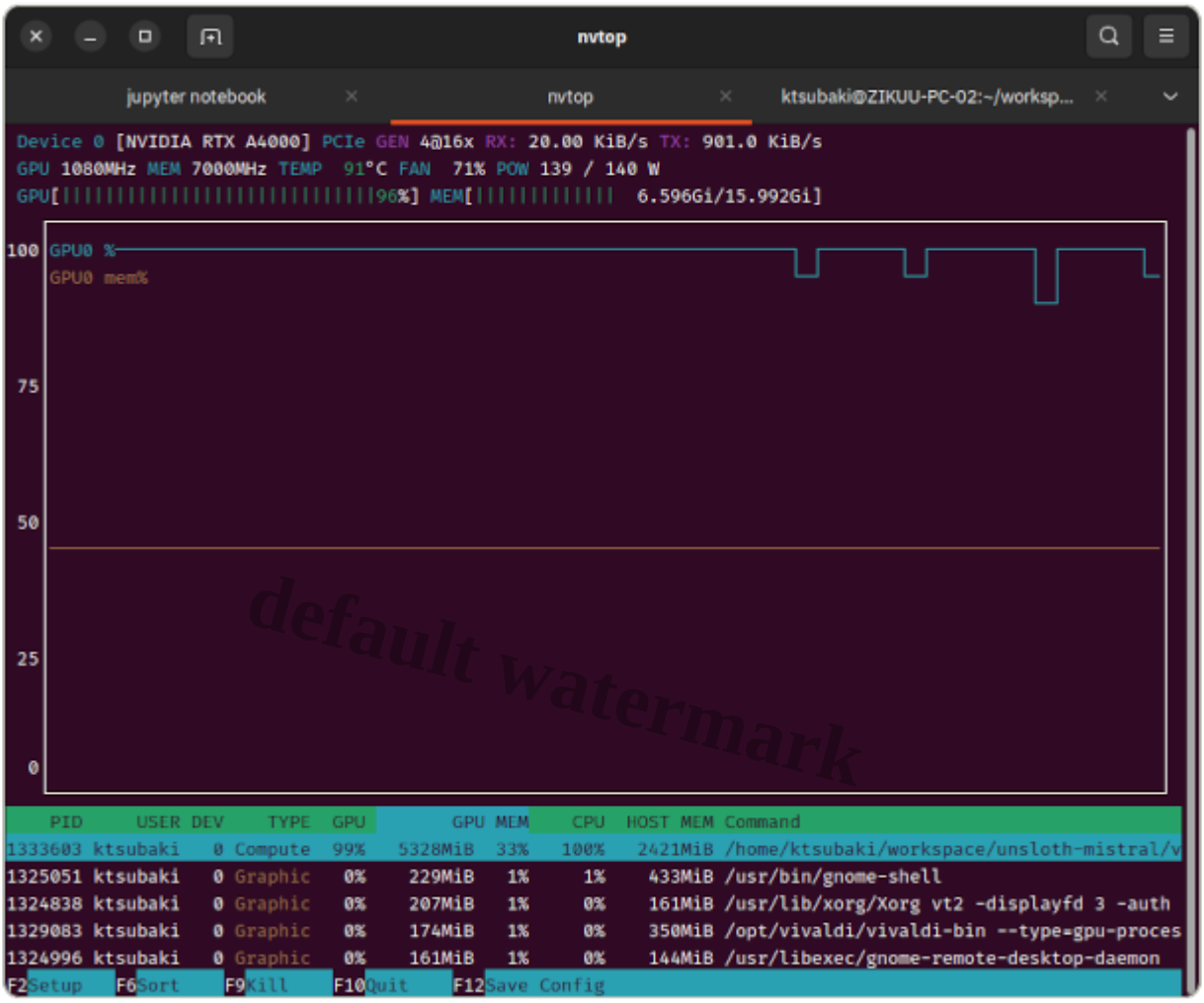
ä»?å??ã•®ã??ã?¼ã??ã•¯ã??Mistral 7Bã??Unslothã??ä½¿ã£ã•¦QLoRAã?•ã?¥ã?¼ã??ã?³ã?°ã??è¡?ã• ?ã?•ã•§ã•?ã??

Unslothã•¯VRAMã®ä½¿ç?¨é?•ã??ç¯?ç´?ã?ã•¦LLMã??ã?•ã?¥ã?¼ã??ã?³ã?°ã•§ã•• ã??ã?©ã?¤ã??ã?©ã?ªã?¼ã•§ã•?ã??



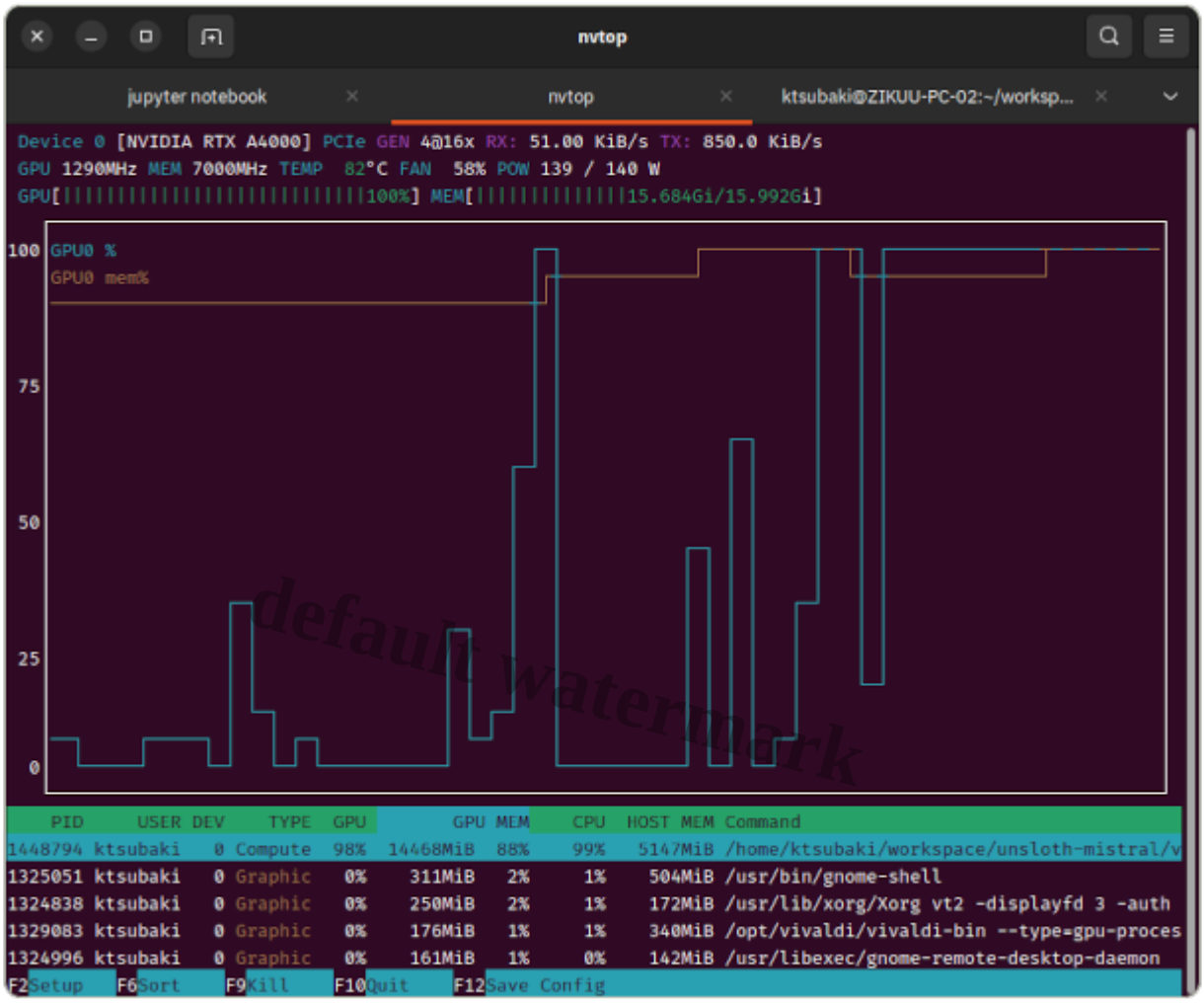
ä»?å??ã•¯VRAMã®ä½¿ç?¨é?•ã•«ç•?ç?®ã•?ã•¦ä½?æ¥ã•?ã•¾ã•?ã•?ã??ä½¿ç?¨ã•?ã•?ã??ã?¼ã?¿ã?»ã??ã??ã•¯ã?ã•®ã??ã?ã?°ã•?ã??æ?½å?ºã•?ã•?ã??ã?¹ã??ã•«å°?ã•?ã??ã?¼ã?¿ã??è¿½å? ã•?ã•?690ä»¶ã•»ã•©ã•®ã??ã•®ã•§ã?•Alpacaå½¢å¼•ã•§ã•?ã??

ã•?ã??ã•¯Mistral 7Bã®4ã??ã??ã??é?•å•å??ã?¢ã??ã?«ã??ã?•ã?¥ã?¼ã??ã?³ã?°ã•?ã•?ã•¨ã••ã•®VRAMä½¿ç?¨é?•ã•§ã•?ã??VRAMä½¿ç?¨é?•ã•¯8GBä»¥å? ã«å•?ã•¾ã•£ã•¦ã•?ã•¾ã•?ã??ã•?ã??ã•ªã??ã?•å¤?å°?é??åº¦ã•?é• ã•?ã•®ã??æ??æ ¢ã•?ã??ã•°ã?•Core i5/Ryzen 5 ã•¨ GeForce RTX 3060 12GBã??çµ?ã•¿å•?ã?•ã?•ã•¦æ¯?è¼?ç??å®?ä¾¡ã«çµ?ã??ã• PCã•§ã??ã??ã•¾ã•?ã??



ã•¡ã•ªã•¿ã•«ã?•Gemma 3 12Bã•®4ã??ã??ã??é?•å•å??ã?¢ã??ã?«ã•§ã??ã?¬ã?¼ã??ã?³ã?°ã•?ã•?ã•¨ã•¯ã•®VRAMä½¿ç?¨é?•ã??ç¢ºèª•ã•?ã•?ã•®ã•?ä¸?ã•®ã?¹ã?¯ã?ªã?¼ã?³ã?·ã?§ã??ã??ã§ã•?ã??

å¡¾ã•®PCã•¯Core i5 13500 + RTX A4000ã®çµ?ã•¿å•?ã?•ã•?ã•§ã•?ã??16GBã•®VRAMã•§ã?®ã?ªã?®ã?ªã??ã??ã??æ??ã•?ã•§ã•?ã??A4000ã•¯æ??æ?°ã®ã?²ã?¼ã? ç?¨GPUã•«æ¯?ã¹ã??ã•¨å?¡ç•?é??åº¦ã•§å?£ã??ã•¾ã•?ã??ã?¿ã?¼ã?³ã?¢ã?©ã?¦ã?³ã??ã??é??ã••ã•?ã??ã•«ã•¯å?ä¾¡æ ¼å¸¯ã•®ã?²ã?¼ã? ç?¨GPUã??ä½¿ã•£ã•?æ?¹ã•?è?¯ã?ã•¨æ?•ã•?ã•¾ã•?ã??ã•?ã•?ã??ã?•ã??ã•£ã•¨é«?é??ã•ªGPUã??æ??ã•«å ¥ã??ã•¦ã?•A4000ã•¯ã?µã?¼ã?ã?¼ã•«ç§»è¨ã•?ã•¦æ?¨è«?å°?ç?¨ã•«ã•?ã•?ã•§ã•?ã??



çµæ??ã•¯æ??å¾ ã?ã•¦ã•?ã•?ã??ã•®ã•«ã•¯ç¨?é• ã•?ã??ã•®ã•«ã•ªã??ã•¾ã•?ã•
?ã??ä»?å¾?ã??è©¦è¡?é?¯èª¤ã??ç¶?ã?ã•¦ã•?ã••ã•¾ã•?ã??æ¬¡ã•®ã??ã?¼ã??ã•
¯ã??ã?¼ã?¿ã?»ã??ã??ã??ç¶ºé°?ã«ã•?ã??ã•?ã•¨ã•¨ã?•ã?•ã?¥ã?¼ã??ã?³ã?°ã??ã?©ã?¡ã?¼ã?¿ã?¼ã•
®èª¿æ?´ã•§ã•?ã??ã??ã•?ã??ã•¯ã?ã•?ã•?ã•?ä»?çµ?ã•¿ã•«ã•ªã•£ã•¦ã•?ã??ã?•ã??ã•?ã??å¤?ã•?ã??ã
•¨ã•?ã•?ã•ªã??ã?•ã•¨ã•?ã•£ã•?ã•?ã•¨ã•?ç•?è§£ã•§ã••ã??ã•¾ã•§ã•¯ã?æ??é??ã•?ã•?ã•?ã•£ã•¦ã??è?¯ã?ã•
®ã•§ã•?ã•¶ã•¨ã••å•?ã??çµ?ã•¿ã•¾ã•?ã??

å•?è??ã•¾ã•§ã•«ã?•ä»?å??å®?è¡?ã•?ã•?ã?³ã?¼ã??ã•§ã•?ã??ã•?ã•?ã•?ã•ªã•¨ã•?ã?•ã•?ã•£ã•?ã??ã•
?æ??æ??ã••ã• ã•?ã•?ã??

```
from unsloth import FastLanguageModel
import torch
from trl import SFTTrainer
from transformers import TrainingArguments, TextStreamer

max_seq_length = 2048
dtype = None
load_in_4bit = True
fourbit_models = [
    "unsloth/mistral-7b-v0.3-bnb-4bit",
```

```
]

model, tokenizer = FastLanguageModel.from_pretrained(
    model_name = "unsloth/mistral-7b-v0.3-bnb-4bit",
    max_seq_length = max_seq_length,
    dtype = dtype,
    load_in_4bit = load_in_4bit,
)

model = FastLanguageModel.get_peft_model(
    model,
    r = 16,
    target_modules = ["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up
    lora_alpha = 16,
    lora_dropout = 0,
    bias = "none",
    use_gradient_checkpointing = True,
    random_state = 3407,
    use_rslora = False,
    loftq_config = None,
)

alpaca_prompt = """Below is an instruction that describes a task, paired with
### Instruction:
{}

### Input:
{}

### Response:
{}"""
EOS_TOKEN = tokenizer.eos_token

def formatting_prompts_func(examples):
    instructions = examples["instruction"]
    inputs = examples["input"]
    outputs = examples["output"]
    texts = []
    for instruction, input, output in zip(instructions, inputs, outputs):
        text = alpaca_prompt.format(instruction, input, output) + EOS_TOKEN
        texts.append(text)
    return { "text": texts, }
pass

from datasets import load_dataset
dataset = load_dataset("json", data_files="datasets/qlora_dataset.json", split
dataset = dataset.map(formatting_prompts_func, batched = True)


training_args = TrainingArguments(
    per_device_train_batch_size = 2,
    gradient_accumulation_steps = 4,
    warmup_steps = 5,
    max_steps = 60,
    learning_rate = 2e-4,
```

default watermark

```
    fp16 = not torch.cuda.is_bf16_supported(),
    lr_scheduler_type = "linear",
    seed = 3407,
    bf16 = torch.cuda.is_bf16_supported(),
    logging_steps = 1,
    optim = "adamw_8bit",
    weight_decay = 0.01,
    output_dir = "outputs"
)

trainer = SFTTrainer(
    model = model,
    tokenizer = tokenizer,
    train_dataset = dataset,
    dataset_text_field = "text",
    max_seq_length = max_seq_length,
    dataset_num_proc = 2,
    packing = False,
    args = training_args
)

trainer_stats = trainer.train()
print(trainer_stats)
trainer.save_model("./trained")
tokenizer.save_pretrained("./trained")

FastLanguageModel.for_inference(model)
inputs = tokenizer(
    [
        alpaca_prompt.format(
            "è³ªå?•ã•«å¯¾ã•?ã•¦è©³ç´°ã•?ã•¤ä¸•å¯§ã•«å??ç?ã•?ã•¦ã••ã• ã•?ã•
?ã??",
            "ã?¢ã??ã•¥ã••ã??å¡¾ZIKUUã•®å¡¾é?·ã•®ã??ã?ã??ã?£ã?¼ã?«",
            "",
        )
    ], return_tensors = "pt").to("cuda")
text_streamer = TextStreamer(tokenizer)
_ = model.generate(**inputs, streamer = text_streamer, max_new_tokens = 128)
```

default watermark

Category

1. æ?¥ã? ã•®æ´»å??

Tags

1. AI
2. Linux
3. LLM
4. Mistral
5. QLoRA
6. Unsloth
7. ã??ã?¡ã?¤ã?³ã?•ã?¥ã?¼ã??ã?³ã?°
8. ä¸?é??å??å¸?

9. å¤§è¦•æ¨¡è¨?èª?ã?¢ã??ã?«
10. å±±æ¢¨ç??

Date Created
2025å¹´8æ??6æ?¥
Author
kazuo-tsubaki

default watermark