```
!pip install datasets transformers accelerate bitsandbytes peft trl
```

```
Requirement already satisfied: datasets in /usr/local/lib/python3.12/dist-packages (4.0.0)
Requirement already satisfied: transformers in /usr/local/lib/python3.12/dist-packages (4.57.1)
Requirement already satisfied: accelerate in /usr/local/lib/python3.12/dist-packages (1.11.0)
Collecting bitsandbytes
  Downloading bitsandbytes-0.48.2-py3-none-manylinux_2_24_x86_64.whl.metadata (10 kB)
Requirement already satisfied: peft in /usr/local/lib/python3.12/dist-packages (0.18.0)
Collecting trl
  Downloading trl-0.25.1-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from datasets) (3.20.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from datasets) (2.0.2)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.12/dist-packages (from datasets) (18.1.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in /usr/local/lib/python3.12/dist-packages (from datasets) (0.3.8)
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in /usr/local/lib/python3.12/dist-packages (from datasets) (2.32.4)
Requirement already satisfied: tqdm>=4.66.3 in /usr/local/lib/python3.12/dist-packages (from datasets) (4.67.1)
Requirement already satisfied: xxhash in /usr/local/lib/python3.12/dist-packages (from datasets) (3.6.0)
Requirement already satisfied: multiprocess<0.70.17 in /usr/local/lib/python3.12/dist-packages (from datasets) (0.70.16)
Requirement already satisfied: fsspec<=2025.3.0,>=2023.1.0 in /usr/local/lib/python3.12/dist-packages (from fsspec[http]<=
Requirement already satisfied: huggingface-hub>=0.24.0 in /usr/local/lib/python3.12/dist-packages (from datasets) (0.36.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.12/dist-packages (from datasets) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from datasets) (6.0.3)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2024.11.6
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.7.0)
Requirement already satisfied: psutil in /usr/local/lib/python3.12/dist-packages (from accelerate) (5.9.5)
Requirement already satisfied: torch>=2.0.0 in /usr/local/lib/python3.12/dist-packages (from accelerate) (2.9.0+cu126)
Requirement already satisfied: aiohttp!=4.0.0a0,!=4.0.0a1 in /usr/local/lib/python3.12/dist-packages (from fsspec[http]<=2
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub>=0.24
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests>=2.32.2-
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests>=2.32.2->datasets) (
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests>=2.32.2->datas
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests>=2.32.2->datas
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (75.2
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (1
Requirement already satisfied: networkx>=2.5.1 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (3.1.6)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch>=2
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0-
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0-
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0
Requirement already satisfied: nvidia-nccl-cu12==2.27.5 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->acc
Requirement already satisfied: nvidia-nvshmem-cu12==3.3.20 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->ac
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0-
Requirement already satisfied: triton==3.5.0 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (3
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas->datasets) (
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas->datasets) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas->datasets) (2025.2)
Requirement already satisfied: aiohappyeyeballs>=2.5.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!
```

```
# core idea (not full script)
from datasets import load_dataset
ds = load_dataset("greengerong/leetcode")
```

```
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning: 
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
```

README.md: 100% 21.0/21.0 [00:00<00:00, 1.81kB/s]

leetcode-train.jsonl: 100% 16.1M/16.1M [00:00<00:00, 17.5MB/s]

Generating train split: 100% 2360/2360 [00:00<00:00, 18857.27 examples/s]

```
ds
```

```
DatasetDict({
    train: Dataset({
        features: ['id', 'slug', 'title', 'difficulty', 'content', 'java', 'c++', 'python', 'javascript'],
```

```
        num_rows: 2360
    })
})
```

```
def make_example(example):
    title = example.get("title", "").strip()
    desc = example.get("content", "").strip()
    solution = example.get("python", "")  # field names may vary; inspect dataset first
    difficulty = example.get("difficulty", "")

    # Compose instruction (short + clean)
    short_desc = "\n".join(desc.splitlines()[0:3])
    instruction = f"Solve the problem: {title}. {short_desc}..."

    # Optional input field (examples, function signature, etc.)
    input_field = example.get("content", "")

    # ------------------------------
    # Elaborated Chain-of-Thought Template
    # ------------------------------
    cot = f"""Chain-of-Thought:

1. **Problem Understanding**
   The problem is titled "{title}".
   The description is: {short_desc}.
   We must understand the required input/output format and identify hidden constraints or edge cases.

2. **Key Observations**
   - Identify crucial patterns in the problem (e.g., sorting behavior, graph structure, DP relation, prefix/suffix propert
   - Infer which constraints determine feasible complexity (O(n), O(n log n), etc.).
   - Detect typical pitfalls (off-by-one, integer overflow, duplicate handling, boundary cases).

3. **Reasoning & Approach Selection**
   - Consider all viable strategies.
   - Discard approaches that violate time or space constraints.
   - Justify the optimal algorithmic paradigm (e.g., hash map, two-pointer, BFS/DFS, DP, binary search, greedy).
   - Explain why the selected approach guarantees correctness.

4. **Algorithm Breakdown**
   - Provide a clear, ordered list of implementation steps.
   - Handle special cases separately (empty input, zero values, duplicates, etc.).
   - Present complexity analysis (time + space).

5. **Dry Run / Mini Simulation**
   - Walk through a representative or tricky example step-by-step.
   - Show how the algorithm transitions through states and reaches the correct output.
   - Confirm correctness and edge-case behavior.

6. **Final Implementation Notes**
   - Mention potential pitfalls to avoid in the final code.
   - Explain any important initialization or boundary conditions.
   - Ensure final code is concise, readable, and correct.

"""

    # Final answer / code
    final = solution if solution else "<write solution here>"

    output = cot + "\nFinal Answer (Python Code):\n```python\n" + final + "\n```"

    return {
        "instruction": instruction,
        "input": input_field,
        "output": output,
        "metadata": {"id": example.get("id"), "difficulty": difficulty}
    }
```

```
x=[]

#iterate over ds and append make_example output in X
# for i in range(len(ds["train"])):
#     x.append(make_example(ds["train"][i]))
# len(x)

for i in range(len(ds["train"])):
    x.append(make_example(ds["train"][i]))
len(x)
```

```
2360
```

```python
print(x[0]["instruction"])
```

````
Solve the problem: Two Sum. Given an array of integers `nums` and an integer `target`, return _indices of the two numbers su

You may assume that each input would have **_exactly_ one solution**, and you may not use the _same_ element twice....
````

```python
print(x[0]["input"])
```

````
Given an array of integers `nums` and an integer `target`, return _indices of the two numbers such that they add up to `targ

You may assume that each input would have **_exactly_ one solution**, and you may not use the _same_ element twice.

You can return the answer in any order.

**Example 1:**

**Input:** nums = \[2,7,11,15\], target = 9
**Output:** \[0,1\]
**Explanation:** Because nums\[0\] + nums\[1\] == 9, we return \[0, 1\].

**Example 2:**

**Input:** nums = \[3,2,4\], target = 6
**Output:** \[1,2\]

**Example 3:**

**Input:** nums = \[3,3\], target = 6
**Output:** \[0,1\]

**Constraints:**

*   `2 <= nums.length <= 104`
*   `-109 <= nums[i] <= 109`
*   `-109 <= target <= 109`
*   **Only one valid answer exists.**

**Follow-up:** Can you come up with an algorithm that is less than `O(n2)` time complexity?
````

```python
print(x[0]["output"])
```

````
Chain-of-Thought:

1. **Problem Understanding**
   The problem is titled "Two Sum".
   The description is: Given an array of integers `nums` and an integer `target`, return _indices of the two numbers such th

You may assume that each input would have **_exactly_ one solution**, and you may not use the _same_ element twice..
   We must understand the required input/output format and identify hidden constraints or edge cases.

2. **Key Observations**
   - Identify crucial patterns in the problem (e.g., sorting behavior, graph structure, DP relation, prefix/suffix propertie
   - Infer which constraints determine feasible complexity (O(n), O(n log n), etc.).
   - Detect typical pitfalls (off-by-one, integer overflow, duplicate handling, boundary cases).

3. **Reasoning & Approach Selection**
   - Consider all viable strategies.
   - Discard approaches that violate time or space constraints.
   - Justify the optimal algorithmic paradigm (e.g., hash map, two-pointer, BFS/DFS, DP, binary search, greedy).
   - Explain why the selected approach guarantees correctness.

4. **Algorithm Breakdown**
   - Provide a clear, ordered list of implementation steps.
   - Handle special cases separately (empty input, zero values, duplicates, etc.).
   - Present complexity analysis (time + space).

5. **Dry Run / Mini Simulation**
   - Walk through a representative or tricky example step-by-step.
   - Show how the algorithm transitions through states and reaches the correct output.
   - Confirm correctness and edge-case behavior.

6. **Final Implementation Notes**
   - Mention potential pitfalls to avoid in the final code.
   - Explain any important initialization or boundary conditions.
   - Ensure final code is concise, readable, and correct.


Final Answer (Python Code):
```python

    ```python
def twoSum(nums, target):
    map = {}
    for i, num in enumerate(nums):
        complement = target - num
        if complement in map:
            return [map[complement], i]
````

```
            map[num] = i
        return []
    ```

        The algorithm leverages a hash map (unordered_map in C++, HashMap in Java, dictionary in Python, and Map in JavaScript).

    This approach has a time complexity of O(n) and a space complexity of O(n) as well.

    ```
```

```
!hf auth login
```

```
    _|    _|  _|    _|    _|_|_|    _|_|_|  _|_|_|  _|      _|    _|_|_|      _|_|_|_|    _|_|      _|_|_|  _|_|_|_|
    _|    _|  _|    _|  _|        _|          _|    _|_|    _|  _|            _|        _|    _|  _|        _|
    _|_|_|_|  _|    _|  _|  _|_|  _|  _|_|    _|    _|  _|  _|  _|  _|_|      _|_|_|    _|_|_|_|  _|        _|_|_|
    _|    _|  _|    _|  _|    _|  _|    _|    _|    _|    _|_|  _|    _|      _|        _|    _|  _|        _|
    _|    _|    _|_|      _|_|_|    _|_|_|  _|_|_|  _|      _|    _|_|_|      _|        _|    _|    _|_|_|  _|_|_|_|

    To log in, `huggingface_hub` requires a token generated from https://huggingface.co/settings/tokens .
Enter your token (input will not be visible):
Add token as git credential? (Y/n) n
Token is valid (permission: fineGrained).
The token `varad_learning` has been saved to /root/.cache/huggingface/stored_tokens
Your token has been saved to /root/.cache/huggingface/token
Login successful.
The current active token is: `varad_learning`
```

```
!pip uninstall -y bitsandbytes # Force uninstall bitsandbytes
!pip install -U bitsandbytes # Reinstall the latest bitsandbytes
```

```
WARNING: Skipping bitsandbytes as it is not installed.
Collecting bitsandbytes
  Downloading bitsandbytes-0.48.2-py3-none-manylinux_2_24_x86_64.whl.metadata (10 kB)
Requirement already satisfied: torch<3,>=2.3 in /usr/local/lib/python3.12/dist-packages (from bitsandbytes) (2.9.0+cu126)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from bitsandbytes) (2.0.2)
Requirement already satisfied: packaging>=20.9 in /usr/local/lib/python3.12/dist-packages (from bitsandbytes) (25.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (3.20.
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bit
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (75.
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (
Requirement already satisfied: networkx>=2.5.1 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (3.1.6)
Requirement already satisfied: fsspec>=0.8.5 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->b
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3-
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3-
Requirement already satisfied: nvidia-nccl-cu12==2.27.5 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bits
Requirement already satisfied: nvidia-nvshmem-cu12==3.3.20 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->b
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bit
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->
Requirement already satisfied: triton==3.5.0 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch<3,>=
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch<3,>=2.3->bitsa
Downloading bitsandbytes-0.48.2-py3-none-manylinux_2_24_x86_64.whl (59.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 59.4/59.4 MB 13.4 MB/s eta 0:00:00
Installing collected packages: bitsandbytes
Successfully installed bitsandbytes-0.48.2
```

```
import accelerate
!pip install -U bitsandbytes
from transformers import AutoTokenizer, AutoModelForCausalLM, BitsAndBytesConfig # Import BitsAndBytesConfig
import torch # Import torch for dtype

model_name = "google/gemma-2-2b-it"   # or gemma-7b-it

tokenizer = AutoTokenizer.from_pretrained(
    model_name,
    trust_remote_code=True
)

# Create a BitsAndBytesConfig object
quantization_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_quant_type="nf4", # Or "fp4"
    bnb_4bit_compute_dtype=torch.bfloat16,
    bnb_4bit_use_double_quant=True,
```

```
)

model = AutoModelForCausalLM.from_pretrained(
    model_name,
    quantization_config=quantization_config, # Pass the quantization_config object
    device_map="auto",
    trust_remote_code=True
)
```

```
Requirement already satisfied: bitsandbytes in /usr/local/lib/python3.12/dist-packages (0.48.2)
Requirement already satisfied: torch<3,>=2.3 in /usr/local/lib/python3.12/dist-packages (from bitsandbytes) (2.9.0+cu126)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from bitsandbytes) (2.0.2)
Requirement already satisfied: packaging>=20.9 in /usr/local/lib/python3.12/dist-packages (from bitsandbytes) (25.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (3.20.
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bit
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (75.
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (
Requirement already satisfied: networkx>=2.5.1 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (3.1.6)
Requirement already satisfied: fsspec>=0.8.5 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->b
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3-
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3-
Requirement already satisfied: nvidia-nccl-cu12==2.27.5 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bits
Requirement already satisfied: nvidia-nvshmem-cu12==3.3.20 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->b
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bit
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->
Requirement already satisfied: triton==3.5.0 in /usr/local/lib/python3.12/dist-packages (from torch<3,>=2.3->bitsandbytes) (
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch<3,>=
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch<3,>=2.3->bitsa
```

model.safetensors.index.json: 100% 24.2k/24.2k [00:00<00:00, 841kB/s]

Fetching 2 files: 100% 2/2 [00:42<00:00, 42.34s/it]

model-00002-of-00002.safetensors: 100% 241M/241M [00:17<00:00, 16.4MB/s]

model-00001-of-00002.safetensors: 100% 4.99G/4.99G [00:42<00:00, 176MB/s]

Loading checkpoint shards: 100% 2/2 [03:21<00:00, 84.91s/it]

generation_config.json: 100% 187/187 [00:00<00:00, 26.4kB/s]

```
tokenizer.model_max_length = 1536
tokenizer.padding_side = "right"
tokenizer.truncation_side = "right"
```

```
from peft import LoraConfig, get_peft_model

lora_config = LoraConfig(
    r=8,                       # very important for T4
    lora_alpha=16,
    lora_dropout=0.05,
    target_modules=["q_proj","k_proj","v_proj","o_proj"],
    bias="none",
    task_type="CAUSAL_LM",
)

model = get_peft_model(model, lora_config)
gradient_checkpointing = True
model.gradient_checkpointing_enable()
# Enable gradient calculation for model inputs, crucial for gradient checkpointing with PEFT
model.enable_input_require_grads()
model.print_trainable_parameters()
```

```
trainable params: 3,194,880 || all params: 2,617,536,768 || trainable%: 0.1221
```

```
from transformers import TrainingArguments

training_args = TrainingArguments(
    output_dir="gemma-leetcode-t4",
    per_device_train_batch_size=1,
    gradient_accumulation_steps=1,
    learning_rate=3e-4,
```

```
    warmup_ratio=0.03,
    num_train_epochs=1,
    logging_steps=20,
    save_steps=5000,
    fp16=True,
    optim="paged_adamw_8bit",
    report_to="none",
    remove_unused_columns=False,
    gradient_checkpointing=True,  # important for T4
)
```

```
# define train_dataset and data_collator
from datasets import Dataset
from transformers import DataCollatorWithPadding

def preprocess_function(examples):
    # examples contains lists under keys "instruction", "input", "output"
    inputs = []
    labels = []

    # build user-only prompts and full chat prompts (vectorized per batch)
    user_prompts = []
    full_prompts = []
    for user_text, inp_text, assistant_text in zip(examples["instruction"],
                                                   examples["input"],
                                                   examples["output"]):
        user = user_text
        if inp_text:
            user = user + "\n\nInput Examples:\n" + inp_text
        # full chat: user + assistant
        chat = [
            {"role": "user", "content": user},
            {"role": "assistant", "content": assistant_text},
        ]

        # make the string prompt using Gemma chat template (no tokenization yet)
        full_prompt = tokenizer.apply_chat_template(chat, tokenize=False)
        # user-only prompt (so we can count tokens to mask labels)
        user_chat = [{"role": "user", "content": user}]
        user_prompt = tokenizer.apply_chat_template(user_chat, tokenize=False)

        full_prompts.append(full_prompt)
        user_prompts.append(user_prompt)

    # tokenize both lists in batch
    tokenized_full = tokenizer(
        full_prompts,
        truncation=True,
        max_length=1536,
        padding=False,
        return_tensors=None
    )

    tokenized_user = tokenizer(
        user_prompts,
        truncation=True,
        max_length=1536,
        padding=False,
        return_tensors=None
    )

    # build labels: copy input_ids, then mask user tokens with -100
    out = {}
    input_ids_list = tokenized_full["input_ids"]
    attention_mask_list = tokenized_full["attention_mask"]
    user_len_list = [len(u) for u in tokenized_user["input_ids"]]

    labels_list = []
    for idx, ids in enumerate(input_ids_list):
        # copy
        lab = ids.copy()
        # mask user tokens (first user_len_list[idx] tokens) to -100
        user_len = user_len_list[idx]
        # safety: ensure we don't exceed
        if user_len > len(lab):
            user_len = len(lab)
        for j in range(user_len):
            lab[j] = -100
        labels_list.append(lab)

    out["input_ids"] = input_ids_list
```

```
    out["attention_mask"] = attention_mask_list
    out["labels"] = labels_list
    return out

# Create the train_dataset (rehash your raw_train_dataset)
raw_train_dataset = Dataset.from_list(x)
train_dataset = raw_train_dataset.map(
    preprocess_function,
    batched=True,
    remove_columns=raw_train_dataset.column_names,
)

# use a simple padding collator (it will also pad labels properly)
data_collator = DataCollatorWithPadding(tokenizer, pad_to_multiple_of=None)
```

Map: 100% 100/100 [00:00<00:00, 126.20 examples/s]

```
from transformers import Trainer

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    data_collator=data_collator,
)
```

```
trainer.train()
```

[100/100 03:26, Epoch 1/1]

Step	Training Loss
20	0.259000
40	0.263800
60	0.258200
80	0.216900
100	0.244900

```
TrainOutput(global_step=100, training_loss=0.24854284286499023, metrics={'train_runtime': 208.1535,
'train_samples_per_second': 0.48, 'train_steps_per_second': 0.48, 'total_flos': 1404243812371968.0, 'train_loss':
0.24854284286499023, 'epoch': 1.0})
```

```
from peft import PeftModel

# Save LoRA adapters only
trainer.model.save_pretrained("gemma-2b-qlora-cot-adapter")
tokenizer.save_pretrained("gemma-2b-qlora-cot-adapter")
```

```
('gemma-2b-qlora-cot-adapter/tokenizer_config.json',
 'gemma-2b-qlora-cot-adapter/special_tokens_map.json',
 'gemma-2b-qlora-cot-adapter/chat_template.jinja',
 'gemma-2b-qlora-cot-adapter/tokenizer.model',
 'gemma-2b-qlora-cot-adapter/added_tokens.json',
 'gemma-2b-qlora-cot-adapter/tokenizer.json')
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving gemma-2b-cot-leetcode.zip to gemma-2b-cot-leetcode.zip
```

```
!unzip gemma-2b-cot-leetcode.zip
```

```
Archive:  gemma-2b-cot-leetcode.zip
   creating: gemma-2b-cot-leetcode/
  inflating: gemma-2b-cot-leetcode/adapter_config.json
  inflating: gemma-2b-cot-leetcode/adapter_model.safetensors
  inflating: gemma-2b-cot-leetcode/chat_template.jinja
  inflating: gemma-2b-cot-leetcode/README.md
  inflating: gemma-2b-cot-leetcode/special_tokens_map.json
  inflating: gemma-2b-cot-leetcode/tokenizer.json
  inflating: gemma-2b-cot-leetcode/tokenizer.model
  inflating: gemma-2b-cot-leetcode/tokenizer_config.json
```

```
from transformers import AutoModelForCausalLM, AutoTokenizer
from peft import PeftModel
```

```
base_model_name = "google/gemma-2-2b-it"   # Changed to match the likely base model of the adapter
lora_path = "gemma-2b-cot-leetcode"

tokenizer = AutoTokenizer.from_pretrained(base_model_name)
base_model = AutoModelForCausalLM.from_pretrained(
    base_model_name,
    torch_dtype="auto",
    device_map="auto"
)

model = PeftModel.from_pretrained(base_model, lora_path)
model.eval()
```

```
tokenizer_config.json: 100%    47.0k/47.0k [00:00<00:00, 5.46MB/s]
tokenizer.model: 100%    4.24M/4.24M [00:00<00:00, 229kB/s]
tokenizer.json: 100%    17.5M/17.5M [00:00<00:00, 34.3MB/s]
special_tokens_map.json: 100%    636/636 [00:00<00:00, 44.4kB/s]
config.json: 100%    838/838 [00:00<00:00, 107kB/s]
model.safetensors.index.json: 100%    24.2k/24.2k [00:00<00:00, 2.63MB/s]
Fetching 2 files: 100%    2/2 [01:24<00:00, 84.47s/it]
model-00001-of-00002.safetensors: 100%    4.99G/4.99G [01:24<00:00, 64.5MB/s]
model-00002-of-00002.safetensors: 100%    241M/241M [00:36<00:00, 6.02MB/s]
Loading checkpoint shards: 100%    2/2 [00:23<00:00, 9.81s/it]
generation_config.json: 100%    187/187 [00:00<00:00, 21.2kB/s]
```

```
PeftModelForCausalLM(
  (base_model): LoraModel(
    (model): Gemma2ForCausalLM(
      (model): Gemma2Model(
        (embed_tokens): Embedding(256000, 2304, padding_idx=0)
        (layers): ModuleList(
          (0-25): 26 x Gemma2DecoderLayer(
            (self_attn): Gemma2Attention(
              (q_proj): lora.Linear(
                (base_layer): Linear(in_features=2304, out_features=2048, bias=False)
                (lora_dropout): ModuleDict(
                  (default): Dropout(p=0.05, inplace=False)
                )
                (lora_A): ModuleDict(
                  (default): Linear(in_features=2304, out_features=8, bias=False)
                )
                (lora_B): ModuleDict(
                  (default): Linear(in_features=8, out_features=2048, bias=False)
                )
                (lora_embedding_A): ParameterDict()
                (lora_embedding_B): ParameterDict()
                (lora_magnitude_vector): ModuleDict()
              )
              (k_proj): lora.Linear(
                (base_layer): Linear(in_features=2304, out_features=1024, bias=False)
                (lora_dropout): ModuleDict(
                  (default): Dropout(p=0.05, inplace=False)
                )
                (lora_A): ModuleDict(
                  (default): Linear(in_features=2304, out_features=8, bias=False)
                )
                (lora_B): ModuleDict(
                  (default): Linear(in_features=8, out_features=1024, bias=False)
                )
                (lora_embedding_A): ParameterDict()
                (lora_embedding_B): ParameterDict()
                (lora_magnitude_vector): ModuleDict()
```

```
def generate(model,tokenizer,instruction,input_text):
  user = instruction
  if input_text:
        user = user + "\n\nInput:\n" + input_text

    # Provide a steering / instruction sentence (safe, short) in the user prompt
  steering = " Please provide a clear step-by-step reasoning labeled 'Step 1:', 'Step 2:', ... followed by the final answer
  user = user + steering

  messages = [{"role": "user", "content": user}]
  inputs = tokenizer.apply_chat_template(
    messages,
    add_generation_prompt=True,
    tokenize=True,
    return_dict=True,
```

```python
    return_tensors="pt",
  ).to(model.device)

  outputs = model.generate(**inputs, max_new_tokens=5000)
  return tokenizer.decode(outputs[0][inputs["input_ids"].shape[-1]:])
```

```
(default): Linear(in_features=8, out_features=2304, bias=False)
```

```python
# ----------------------------
# Inference on a random example
# ----------------------------
import random

idx = random.randint(0, len(x)-1)
test = x[idx]

print("\n⭐ TEST EXAMPLE:")
print(test["instruction"])

result = generate(
    model,
    tokenizer,
    instruction=test["instruction"],
    input_text=test["input"]
)

print("\n⭐ MODEL OUTPUT:")
print(result)

print("\n⭐ EXPECTED OUTPUT:")
print(test["output"])
```

````
⭐ TEST EXAMPLE:
Solve the problem: Spiral Matrix II. Given a positive integer `n`, generate an `n x n` `matrix` filled with elements from

**Example 1:**...

⭐ MODEL OUTPUT:
## Solution:

**Step 1:** We will use a 2D array to store the matrix. The size of the matrix will be `n x n`.

**Step 2:** We will iterate through the matrix in a spiral order. The spiral order is defined as follows:

*   Start at the top-left corner of the matrix.
*   Move right until you reach the right edge of the matrix.
*   Move down until you reach the bottom edge of the matrix.
*   Move left until you reach the left edge of the matrix.
*   Move up until you reach the top edge of the matrix.
*   Repeat the process.

**Step 3:** We will use a counter to keep track of the current element in the matrix.

**Step 4:** We will use a nested loop to iterate through the matrix.

**Step 5:** We will increment the counter and add the current element to the matrix.

**Step 6:** We will decrement the counter and add the current element to the matrix.

**Step 7:** We will repeat the process until we have filled the entire matrix.

**Step 8:** We will return the matrix.

**Code:**

```python
def generateMatrix(n):
    matrix = [[0 for _ in range(n)] for _ in range(n)]
    counter = 1
    top, bottom = 0, n - 1
    left, right = 0, n - 1

    while top <= bottom and left <= right:
        for i in range(left, right + 1):
            matrix[top][i] = counter
            counter += 1
        top += 1
        for i in range(top, bottom + 1):
            matrix[i][right] = counter
            counter += 1
        right -= 1
        if top <= bottom and left <= right:
            for i in range(right, left - 1, -1):
                matrix[bottom][i] = counter
                counter += 1
````
````

```
bottom -= 1
for i in range(bottom, top - 1, -1):
    matrix[i][left] = counter
```