2024_National Institute of the Korean Language AI Malpyeong Competition

June 22, 2025

[ ]:

```
!pip install torch
!pip install transformers
!pip install peft
!pip install trl
!pip install -U bitsandbytes
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-packages
(2.3.1+cu121)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from torch) (3.15.4)
Requirement already satisfied: typing-extensions>=4.8.0 in
/usr/local/lib/python3.10/dist-packages (from torch) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages
(from torch) (1.13.2)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-
packages (from torch) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages
(from torch) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages
(from torch) (2024.6.1)
Collecting nvidia-cuda-nvrtc-cu12==12.1.105 (from torch)
  Using cached nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-
manylinux1_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-runtime-cu12==12.1.105 (from torch)
  Using cached nvidia_cuda_runtime_cu12-12.1.105-py3-none-
manylinux1_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-cupti-cu12==12.1.105 (from torch)
  Using cached nvidia_cuda_cupti_cu12-12.1.105-py3-none-
manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cudnn-cu12==8.9.2.26 (from torch)
  Using cached nvidia_cudnn_cu12-8.9.2.26-py3-none-
manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cublas-cu12==12.1.3.1 (from torch)
  Using cached nvidia_cublas_cu12-12.1.3.1-py3-none-
manylinux1_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cufft-cu12==11.0.2.54 (from torch)
```

```
  Using cached nvidia_cufft_cu12-11.0.2.54-py3-none-
manylinux1_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-curand-cu12==10.3.2.106 (from torch)
  Using cached nvidia_curand_cu12-10.3.2.106-py3-none-
manylinux1_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cusolver-cu12==11.4.5.107 (from torch)
  Using cached nvidia_cusolver_cu12-11.4.5.107-py3-none-
manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cusparse-cu12==12.1.0.106 (from torch)
  Using cached nvidia_cusparse_cu12-12.1.0.106-py3-none-
manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-nccl-cu12==2.20.5 (from torch)
  Using cached nvidia_nccl_cu12-2.20.5-py3-none-
manylinux2014_x86_64.whl.metadata (1.8 kB)
Collecting nvidia-nvtx-cu12==12.1.105 (from torch)
  Using cached nvidia_nvtx_cu12-12.1.105-py3-none-manylinux1_x86_64.whl.metadata
(1.7 kB)
Requirement already satisfied: triton==2.3.1 in /usr/local/lib/python3.10/dist-
packages (from torch) (2.3.1)
Collecting nvidia-nvjitlink-cu12 (from nvidia-cusolver-cu12==11.4.5.107->torch)
  Using cached nvidia_nvjitlink_cu12-12.6.20-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch) (2.1.5)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from sympy->torch) (1.3.0)
Using cached nvidia_cublas_cu12-12.1.3.1-py3-none-manylinux1_x86_64.whl (410.6
MB)
Using cached nvidia_cuda_cupti_cu12-12.1.105-py3-none-manylinux1_x86_64.whl
(14.1 MB)
Using cached nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-manylinux1_x86_64.whl
(23.7 MB)
Using cached nvidia_cuda_runtime_cu12-12.1.105-py3-none-manylinux1_x86_64.whl
(823 kB)
Using cached nvidia_cudnn_cu12-8.9.2.26-py3-none-manylinux1_x86_64.whl (731.7
MB)
Using cached nvidia_cufft_cu12-11.0.2.54-py3-none-manylinux1_x86_64.whl (121.6
MB)
Using cached nvidia_curand_cu12-10.3.2.106-py3-none-manylinux1_x86_64.whl (56.5
MB)
Using cached nvidia_cusolver_cu12-11.4.5.107-py3-none-manylinux1_x86_64.whl
(124.2 MB)
Using cached nvidia_cusparse_cu12-12.1.0.106-py3-none-manylinux1_x86_64.whl
(196.0 MB)
Using cached nvidia_nccl_cu12-2.20.5-py3-none-manylinux2014_x86_64.whl (176.2
MB)
Using cached nvidia_nvtx_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (99 kB)
Using cached nvidia_nvjitlink_cu12-12.6.20-py3-none-manylinux2014_x86_64.whl
```

```
(19.7 MB)
Installing collected packages: nvidia-nvtx-cu12, nvidia-nvjitlink-cu12, nvidia-
nccl-cu12, nvidia-curand-cu12, nvidia-cufft-cu12, nvidia-cuda-runtime-cu12,
nvidia-cuda-nvrtc-cu12, nvidia-cuda-cupti-cu12, nvidia-cublas-cu12, nvidia-
cusparse-cu12, nvidia-cudnn-cu12, nvidia-cusolver-cu12
Successfully installed nvidia-cublas-cu12-12.1.3.1 nvidia-cuda-cupti-
cu12-12.1.105 nvidia-cuda-nvrtc-cu12-12.1.105 nvidia-cuda-runtime-cu12-12.1.105
nvidia-cudnn-cu12-8.9.2.26 nvidia-cufft-cu12-11.0.2.54 nvidia-curand-
cu12-10.3.2.106 nvidia-cusolver-cu12-11.4.5.107 nvidia-cusparse-cu12-12.1.0.106
nvidia-nccl-cu12-2.20.5 nvidia-nvjitlink-cu12-12.6.20 nvidia-nvtx-cu12-12.1.105
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-
packages (4.42.4)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from transformers) (3.15.4)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.23.5)
Requirement already satisfied: numpy<2.0,>=1.17 in
/usr/local/lib/python3.10/dist-packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from transformers) (24.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.10/dist-packages (from transformers) (2024.5.15)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from transformers) (2.32.3)
Requirement already satisfied: safetensors>=0.4.1 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.4.4)
Requirement already satisfied: tokenizers<0.20,>=0.19 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.19.1)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-
packages (from transformers) (4.66.5)
Requirement already satisfied: fsspec>=2023.5.0 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.23.2->transformers) (2024.6.1)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.23.2->transformers) (4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->transformers) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (2024.7.4)
Collecting peft
  Downloading peft-0.12.0-py3-none-any.whl.metadata (13 kB)
```

```
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from peft) (1.26.4)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from peft) (24.1)
Requirement already satisfied: psutil in /usr/local/lib/python3.10/dist-packages
(from peft) (5.9.5)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.10/dist-packages
(from peft) (6.0.2)
Requirement already satisfied: torch>=1.13.0 in /usr/local/lib/python3.10/dist-
packages (from peft) (2.3.1+cu121)
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-
packages (from peft) (4.42.4)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages
(from peft) (4.66.5)
Requirement already satisfied: accelerate>=0.21.0 in
/usr/local/lib/python3.10/dist-packages (from peft) (0.32.1)
Requirement already satisfied: safetensors in /usr/local/lib/python3.10/dist-
packages (from peft) (0.4.4)
Requirement already satisfied: huggingface-hub>=0.17.0 in
/usr/local/lib/python3.10/dist-packages (from peft) (0.23.5)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from huggingface-hub>=0.17.0->peft) (3.15.4)
Requirement already satisfied: fsspec>=2023.5.0 in
/usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.17.0->peft)
(2024.6.1)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from huggingface-hub>=0.17.0->peft) (2.32.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.17.0->peft)
(4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages
(from torch>=1.13.0->peft) (1.13.2)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-
packages (from torch>=1.13.0->peft) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages
(from torch>=1.13.0->peft) (3.1.4)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (12.1.105)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (12.1.105)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (12.1.105)
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (8.9.2.26)
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (12.1.3.1)
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (11.0.2.54)
```

```
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (10.3.2.106)
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.20.5 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (2.20.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (12.1.105)
Requirement already satisfied: triton==2.3.1 in /usr/local/lib/python3.10/dist-
packages (from torch>=1.13.0->peft) (2.3.1)
Requirement already satisfied: nvidia-nvjitlink-cu12 in
/usr/local/lib/python3.10/dist-packages (from nvidia-cusolver-
cu12==11.4.5.107->torch>=1.13.0->peft) (12.6.20)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.10/dist-packages (from transformers->peft) (2024.5.15)
Requirement already satisfied: tokenizers<0.20,>=0.19 in
/usr/local/lib/python3.10/dist-packages (from transformers->peft) (0.19.1)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.13.0->peft)
(2.1.5)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->huggingface-
hub>=0.17.0->peft) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->huggingface-hub>=0.17.0->peft) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->huggingface-
hub>=0.17.0->peft) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->huggingface-
hub>=0.17.0->peft) (2024.7.4)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.13.0->peft)
(1.3.0)
Downloading peft-0.12.0-py3-none-any.whl (296 kB)
                         296.4/296.4 kB
19.0 MB/s eta 0:00:00
Installing collected packages: peft
Successfully installed peft-0.12.0
Collecting trl
  Downloading trl-0.9.6-py3-none-any.whl.metadata (12 kB)
Requirement already satisfied: torch>=1.4.0 in /usr/local/lib/python3.10/dist-
packages (from trl) (2.3.1+cu121)
Requirement already satisfied: transformers>=4.31.0 in
/usr/local/lib/python3.10/dist-packages (from trl) (4.42.4)
Requirement already satisfied: numpy<2.0.0,>=1.18.2 in
```

```
/usr/local/lib/python3.10/dist-packages (from trl) (1.26.4)
Requirement already satisfied: accelerate in /usr/local/lib/python3.10/dist-
packages (from trl) (0.32.1)
Collecting datasets (from trl)
  Downloading datasets-2.21.0-py3-none-any.whl.metadata (21 kB)
Collecting tyro>=0.5.11 (from trl)
  Downloading tyro-0.8.8-py3-none-any.whl.metadata (8.4 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from torch>=1.4.0->trl) (3.15.4)
Requirement already satisfied: typing-extensions>=4.8.0 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages
(from torch>=1.4.0->trl) (1.13.2)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-
packages (from torch>=1.4.0->trl) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages
(from torch>=1.4.0->trl) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages
(from torch>=1.4.0->trl) (2024.6.1)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (12.1.105)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (12.1.105)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (12.1.105)
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (8.9.2.26)
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (12.1.3.1)
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (11.0.2.54)
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (10.3.2.106)
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.20.5 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (2.20.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.4.0->trl) (12.1.105)
Requirement already satisfied: triton==2.3.1 in /usr/local/lib/python3.10/dist-
packages (from torch>=1.4.0->trl) (2.3.1)
Requirement already satisfied: nvidia-nvjitlink-cu12 in
/usr/local/lib/python3.10/dist-packages (from nvidia-cusolver-
cu12==11.4.5.107->torch>=1.4.0->trl) (12.6.20)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in
/usr/local/lib/python3.10/dist-packages (from transformers>=4.31.0->trl)
```

```
(0.23.5)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from transformers>=4.31.0->trl) (24.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from transformers>=4.31.0->trl) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.10/dist-packages (from transformers>=4.31.0->trl)
(2024.5.15)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from transformers>=4.31.0->trl) (2.32.3)
Requirement already satisfied: safetensors>=0.4.1 in
/usr/local/lib/python3.10/dist-packages (from transformers>=4.31.0->trl) (0.4.4)
Requirement already satisfied: tokenizers<0.20,>=0.19 in
/usr/local/lib/python3.10/dist-packages (from transformers>=4.31.0->trl)
(0.19.1)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-
packages (from transformers>=4.31.0->trl) (4.66.5)
Requirement already satisfied: docstring-parser>=0.16 in
/usr/local/lib/python3.10/dist-packages (from tyro>=0.5.11->trl) (0.16)
Requirement already satisfied: rich>=11.1.0 in /usr/local/lib/python3.10/dist-
packages (from tyro>=0.5.11->trl) (13.7.1)
Collecting shtab>=1.5.6 (from tyro>=0.5.11->trl)
  Downloading shtab-1.7.1-py3-none-any.whl.metadata (7.3 kB)
Requirement already satisfied: psutil in /usr/local/lib/python3.10/dist-packages
(from accelerate->trl) (5.9.5)
Collecting pyarrow>=15.0.0 (from datasets->trl)
  Downloading pyarrow-17.0.0-cp310-cp310-manylinux_2_28_x86_64.whl.metadata (3.3
kB)
Collecting dill<0.3.9,>=0.3.0 (from datasets->trl)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(from datasets->trl) (2.1.4)
Collecting xxhash (from datasets->trl)
  Downloading
xxhash-3.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(12 kB)
Collecting multiprocess (from datasets->trl)
  Downloading multiprocess-0.70.16-py310-none-any.whl.metadata (7.2 kB)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.10/dist-
packages (from datasets->trl) (3.10.5)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets->trl) (2.4.0)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets->trl) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-
packages (from aiohttp->datasets->trl) (24.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets->trl) (1.4.1)
```

```
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets->trl) (6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-
packages (from aiohttp->datasets->trl) (1.9.4)
Requirement already satisfied: async-timeout<5.0,>=4.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets->trl) (4.0.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from
requests->transformers>=4.31.0->trl) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->transformers>=4.31.0->trl) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from
requests->transformers>=4.31.0->trl) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from
requests->transformers>=4.31.0->trl) (2024.7.4)
Requirement already satisfied: markdown-it-py>=2.2.0 in
/usr/local/lib/python3.10/dist-packages (from rich>=11.1.0->tyro>=0.5.11->trl)
(3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in
/usr/local/lib/python3.10/dist-packages (from rich>=11.1.0->tyro>=0.5.11->trl)
(2.16.1)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.4.0->trl) (2.1.5)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas->datasets->trl) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas->datasets->trl) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-
packages (from pandas->datasets->trl) (2024.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.4.0->trl) (1.3.0)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-
packages (from markdown-it-py>=2.2.0->rich>=11.1.0->tyro>=0.5.11->trl) (0.1.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.2->pandas->datasets->trl) (1.16.0)
Downloading trl-0.9.6-py3-none-any.whl (245 kB)
                         245.8/245.8 kB
14.1 MB/s eta 0:00:00
Downloading tyro-0.8.8-py3-none-any.whl (104 kB)
                         104.6/104.6 kB
9.9 MB/s eta 0:00:00
Downloading datasets-2.21.0-py3-none-any.whl (527 kB)
                         527.3/527.3 kB
26.5 MB/s eta 0:00:00
Downloading dill-0.3.8-py3-none-any.whl (116 kB)
                         116.3/116.3 kB
```

```
11.5 MB/s eta 0:00:00
Downloading pyarrow-17.0.0-cp310-cp310-manylinux_2_28_x86_64.whl (39.9 MB)
                         39.9/39.9 MB
55.9 MB/s eta 0:00:00
Downloading shtab-1.7.1-py3-none-any.whl (14 kB)
Downloading multiprocess-0.70.16-py310-none-any.whl (134 kB)
                         134.8/134.8 kB
12.1 MB/s eta 0:00:00
Downloading
xxhash-3.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
                         194.1/194.1 kB
18.0 MB/s eta 0:00:00
Installing collected packages: xxhash, shtab, pyarrow, dill, multiprocess,
tyro, datasets, trl
  Attempting uninstall: pyarrow
    Found existing installation: pyarrow 14.0.2
    Uninstalling pyarrow-14.0.2:
      Successfully uninstalled pyarrow-14.0.2
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
cudf-cu12 24.4.1 requires pyarrow<15.0.0a0,>=14.0.1, but you have pyarrow 17.0.0
which is incompatible.
ibis-framework 8.0.0 requires pyarrow<16,>=2, but you have pyarrow 17.0.0 which
is incompatible.
Successfully installed datasets-2.21.0 dill-0.3.8 multiprocess-0.70.16
pyarrow-17.0.0 shtab-1.7.1 trl-0.9.6 tyro-0.8.8 xxhash-3.5.0
Collecting bitsandbytes
  Downloading bitsandbytes-0.43.3-py3-none-manylinux_2_24_x86_64.whl.metadata
(3.5 kB)
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-packages
(from bitsandbytes) (2.3.1+cu121)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages
(from bitsandbytes) (1.26.4)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from torch->bitsandbytes) (3.15.4)
Requirement already satisfied: typing-extensions>=4.8.0 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages
(from torch->bitsandbytes) (1.13.2)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-
packages (from torch->bitsandbytes) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages
(from torch->bitsandbytes) (3.1.4)
```

```
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages
(from torch->bitsandbytes) (2024.6.1)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (12.1.105)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (12.1.105)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (12.1.105)
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (8.9.2.26)
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (12.1.3.1)
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (11.0.2.54)
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (10.3.2.106)
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.20.5 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (2.20.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch->bitsandbytes) (12.1.105)
Requirement already satisfied: triton==2.3.1 in /usr/local/lib/python3.10/dist-
packages (from torch->bitsandbytes) (2.3.1)
Requirement already satisfied: nvidia-nvjitlink-cu12 in
/usr/local/lib/python3.10/dist-packages (from nvidia-cusolver-
cu12==11.4.5.107->torch->bitsandbytes) (12.6.20)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch->bitsandbytes)
(2.1.5)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from sympy->torch->bitsandbytes)
(1.3.0)
Downloading bitsandbytes-0.43.3-py3-none-manylinux_2_24_x86_64.whl (137.5 MB)
                         137.5/137.5 MB
16.8 MB/s eta 0:00:00
Installing collected packages: bitsandbytes
Successfully installed bitsandbytes-0.43.3
```

[ ]:

```python
import argparse
import numpy as np
import torch
from datasets import Dataset
from transformers import AutoModelForCausalLM, AutoTokenizer, AutoConfig
from trl import SFTTrainer, SFTConfig
from transformers import BitsAndBytesConfig
from accelerate import init_empty_weights
from peft import LoraConfig

from peft import PeftModel
import json
import gc
import torch
from torch.utils.data import Dataset
```

[ ]:

```
#     json
output="./result_ensemble.json"
```

v1

[ ]:

```
Category_Array = []
class CustomDataset(Dataset):
    def __init__(self, fname, tokenizer):
        IGNORE_INDEX=-100
        self.inp = []
        self.trg = []
        self.label = []

        PROMPT = '''You are a helpful AI assistant. Please answer the user's␣
↪questions correctly.      AI       .                .'''
        answer_dict = {
            "": None,
            "inference_1": 0,
            "inference_2": 1,
            "inference_3": 2
        }

        with open(fname, "r") as f:
            data = json.load(f)

        def make_chat(inp):
            chat = ["[Conversation]"]
            for cvt in inp['conversation']:
                speaker = cvt['speaker']
                utterance = cvt['utterance']
                chat.append(f" {speaker}: {utterance}   {speaker}  ")
            chat = "\n".join(chat)

            question = f"[Question]\n    {inp['category']}"
            if (ord(inp['category'][-1]) - ord(" ")) % 28 > 0:
                question += "  "
            else:
```

```
            question += " "

        question += "     ?"

        chat = chat + "\n\n" + question + "\n\n[Option]\n"
        chat += f"A. {inp['inference_1']}\n"
        chat += f"B. {inp['inference_2']}\n"
        chat += f"C. {inp['inference_3']}"

        ### CoT   ###
        cot_guidance = '''
                       :
        1.          .
        2.           .
        3.     (A, B, C)               .
        4.              (A, B, C)       .'''

        chat = chat + "\n\n" + cot_guidance

        return chat

    for example in data:
        chat = make_chat(example["input"])
        message = [
            {"role": "system", "content": PROMPT},
            {"role": "user", "content": chat},
        ]

        source = tokenizer.apply_chat_template(
            message,
            add_generation_prompt=True,
            return_tensors="pt",
        )

        target = ""
        if example["output"] == "inference_1":
            target = f"A. {example['input']['inference_1']}{tokenizer.
↪eos_token}"
        elif example["output"] == "inference_2":
            target = f"B. {example['input']['inference_2']}{tokenizer.
↪eos_token}"
        elif example["output"] == "inference_3":
            target = f"C. {example['input']['inference_3']}{tokenizer.
↪eos_token}"

        target = tokenizer(target,
                  return_attention_mask=False,
                      add_special_tokens=False,
                      return_tensors="pt")
            target["input_ids"] = target["input_ids"].type(torch.int64)

            input_ids = torch.concat((source[0], target["input_ids"][0]))
            labels = torch.concat((torch.LongTensor([IGNORE_INDEX] * source[0].
 ↪shape[0]), target["input_ids"][0]))

            self.inp.append(input_ids)
            self.label.append(labels)
            self.trg.append(answer_dict[example["output"]])

    def __len__(self):
        return len(self.inp)

    def __getitem__(self, idx):
        return self.inp[idx], self.trg[idx]


class DataCollatorForSupervisedDataset(object):
    def __init__(self, tokenizer):
        self.tokenizer = tokenizer

    def __call__(self, instances):
        input_ids, labels = tuple([instance[key] for instance in instances] for␣
 ↪key in ("input_ids", "labels"))
        input_ids = torch.nn.utils.rnn.pad_sequence(
            [torch.tensor(ids) for ids in input_ids], batch_first=True,␣
 ↪padding_value=self.tokenizer.pad_token_id
        )
        labels = torch.nn.utils.rnn.pad_sequence([torch.tensor(lbls) for lbls␣
 ↪in labels], batch_first=True, padding_value=-100)
        return dict(
            input_ids=input_ids,
            labels=labels,
            attention_mask=input_ids.ne(self.tokenizer.pad_token_id),
        )
```

[ ]:

```
output_model_path ="./model_path/results_1"
```

[ ]:

```
bnb_config = BitsAndBytesConfig(        # 4-bit quantization
        load_in_4bit=True,
        bnb_4bit_quant_type="nf4",
        bnb_4bit_compute_dtype=torch.bfloat16
    )

torch.cuda.empty_cache()
```

```
gc.collect()

model = AutoModelForCausalLM.from_pretrained(
    "chihoonlee10/T3Q-ko-solar-dpo-v7.0",
    device_map="auto",
    quantization_config = bnb_config
)

tokenizer = AutoTokenizer.from_pretrained("chihoonlee10/T3Q-ko-solar-dpo-v7.0")
tokenizer.pad_token = tokenizer.eos_token

gc.collect()

# LoRA
model = PeftModel.from_pretrained(model, output_model_path)
model.eval()
print("model loaded")
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:89:
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 it as secret in your Google Colab
and restart your session.
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(
```

```
config.json:   0%|          | 0.00/710 [00:00<?, ?B/s]
```

```
model.safetensors.index.json:   0%|          | 0.00/35.8k [00:00<?, ?B/s]
```

```
Downloading shards:   0%|          | 0/5 [00:00<?, ?it/s]
```

```
model-00001-of-00005.safetensors:   0%|          | 0.00/4.94G [00:00<?, ?B/s]
```

```
model-00002-of-00005.safetensors:   0%|          | 0.00/5.00G [00:00<?, ?B/s]
```

```
model-00003-of-00005.safetensors:   0%|          | 0.00/4.92G [00:00<?, ?B/s]
```

```
model-00004-of-00005.safetensors:   0%|          | 0.00/4.92G [00:00<?, ?B/s]
```

```
model-00005-of-00005.safetensors:   0%|          | 0.00/1.69G [00:00<?, ?B/s]
```

```
Loading checkpoint shards:   0%|          | 0/5 [00:00<?, ?it/s]
```

```
generation_config.json:   0%|          | 0.00/154 [00:00<?, ?B/s]
```

```
tokenizer_config.json:   0%|          | 0.00/1.03k [00:00<?, ?B/s]
```

```
tokenizer.json:   0%|          | 0.00/1.80M [00:00<?, ?B/s]
```

```
special_tokens_map.json:   0%|          | 0.00/551 [00:00<?, ?B/s]
```

```
model loaded
```

[ ]:

```python
import tqdm
import numpy

dataset = CustomDataset("./dataset/        _    /    _test.json", tokenizer)

answer_dict = {
    0: "inference_1",
    1: "inference_2",
    2: "inference_3",
}

with open("./dataset/        _    /    _test.json", "r") as f:
    result = json.load(f)

with torch.no_grad():
    prob_array = []
    for idx in tqdm.tqdm(range(len(dataset))):
        inp, _ = dataset[idx]
        outputs = model(
            inp.to("cuda:0").unsqueeze(0)
        )
        logits = outputs.logits[:,-1].flatten()
        probs = (
            torch.nn.functional.softmax(
                torch.tensor(
                    [
                        logits[tokenizer.vocab['A']],
                        logits[tokenizer.vocab['B']],
                        logits[tokenizer.vocab['C']],
                    ]
                ),
                dim=0,
            )
            .detach()
            .cpu()
            .to(torch.float32)
            .numpy()
        )

        #     (    )
        prob_array.append(probs)

        torch.cuda.empty_cache()
        gc.collect()
```

```
No chat template is set for this tokenizer, falling back to a default class-
```

```
level template. This is very error-prone, because models are often trained with
templates different from the class default! Default chat templates are a legacy
feature and will be removed in Transformers v4.43, at which point any code
depending on them will stop working. We recommend setting a valid chat template
before then to ensure that this model continues working without issues.
100%|      | 605/605 [12:18<00:00,  1.22s/it]
```

[ ]:

```
rounded_arrays = [np.round(arr, 3) for arr in prob_array]
```

v2

[ ]:

```
output_model_path ="./model_path/results_2"
```

[ ]:

```
bnb_config = BitsAndBytesConfig(        # 4-bit quantization
        load_in_4bit=True,
        bnb_4bit_quant_type="nf4",
        bnb_4bit_compute_dtype=torch.bfloat16
    )

torch.cuda.empty_cache()
gc.collect()

model = AutoModelForCausalLM.from_pretrained(
    "chihoonlee10/T3Q-ko-solar-dpo-v7.0",
    device_map="auto",
    quantization_config = bnb_config
)

tokenizer = AutoTokenizer.from_pretrained("chihoonlee10/T3Q-ko-solar-dpo-v7.0")
tokenizer.pad_token = tokenizer.eos_token

gc.collect()

# LoRA
model = PeftModel.from_pretrained(model, output_model_path)
model.eval()
print("model loaded")
```

```
Loading checkpoint shards:   0%|          | 0/5 [00:00<?, ?it/s]
```

```
model loaded
```

[ ]:

```
##   chat prompt ##
class CustomDataset(Dataset):
    def __init__(self, fname, tokenizer):
        IGNORE_INDEX=-100
        self.inp = []
        self.trg = []
        self.label = []

        PROMPT = '''You are a helpful AI assistant. Please answer the user's␣
↪questions kindly.      AI       .                .'''
        answer_dict = {
            "": None,
            "inference_1": 0,
            "inference_2": 1,
            "inference_3": 2
        }

        with open(fname, "r") as f:
            data = json.load(f)

        def make_chat(inp):
            chat = ["[Conversation]"]
            for cvt in inp['conversation']:
                speaker = cvt['speaker']
                utterance = cvt['utterance']
                utterance_id = cvt['utterance_id']
                chat.append(f" {speaker} [{utterance_id}]: {utterance}")
            chat = "\n".join(chat)

            utterance_array = []
            for utterance_id in inp['reference_id']:
                utterance_array.append(utterance_id)  #conversation␣
↪utterance_id

            question = f"[Question]\n    {utterance_array}    {inp['category']}"
            if (ord(inp['category'][-1]) - ord(" ")) % 28 > 0:
                question += "  "
            else:
                question += " "

            question += "      ?"

            chat = chat + "\n\n" + question + "\n\n[Option]\n"
            chat += f"A. {inp['inference_1']}\n"
            chat += f"B. {inp['inference_2']}\n"
            chat += f"C. {inp['inference_3']}"

            return chat

        for example in data:
            chat = make_chat(example["input"])
            message = [
                {"role": "system", "content": PROMPT},
                {"role": "user", "content": chat},
            ]
            source = tokenizer.apply_chat_template(
                message,
                add_generation_prompt=True,
                return_tensors="pt",
            )

            target = ""
            if example["output"] == "inference_1":
                target = f"A. {example['input']['inference_1']}{tokenizer.
↪eos_token}"
            elif example["output"] == "inference_2":
                target = f"B. {example['input']['inference_2']}{tokenizer.
↪eos_token}"
            elif example["output"] == "inference_3":
                target = f"C. {example['input']['inference_3']}{tokenizer.
↪eos_token}"

            target = tokenizer(target,
                      return_attention_mask=False,
                      add_special_tokens=False,
                      return_tensors="pt")
            target["input_ids"] = target["input_ids"].type(torch.int64)

            input_ids = torch.concat((source[0], target["input_ids"][0]))
            labels = torch.concat((torch.LongTensor([IGNORE_INDEX] * source[0].
↪shape[0]), target["input_ids"][0]))

            self.inp.append(input_ids)
            self.label.append(labels)
            self.trg.append(answer_dict[example["output"]])

    def __len__(self):
        return len(self.inp)

    def __getitem__(self, idx):
        return self.inp[idx], self.trg[idx]
```

[ ]:

```
import tqdm
import numpy

dataset = CustomDataset("./dataset/        _   /   _test.json", tokenizer)

answer_dict = {
    0: "inference_1",
    1: "inference_2",
    2: "inference_3",
}

with open("./dataset/        _   /   _test.json", "r") as f:
    result_2 = json.load(f)

with torch.no_grad():
    prob_array_2 = []
    for idx in tqdm.tqdm(range(len(dataset))):
        inp, _ = dataset[idx]
        outputs = model(
            inp.to("cuda:0").unsqueeze(0)
        )
        logits = outputs.logits[:,-1].flatten()
        probs = (
            torch.nn.functional.softmax(
                torch.tensor(
                    [
                        logits[tokenizer.vocab['A']],
                        logits[tokenizer.vocab['B']],
                        logits[tokenizer.vocab['C']],
                    ]
                ),
                dim=0,
            )
            .detach()
            .cpu()
            .to(torch.float32)
            .numpy()
        )

        #     (    )
        prob_array_2.append(probs)

        torch.cuda.empty_cache()
        gc.collect()
```

```
100%|      | 605/605 [13:16<00:00,  1.32s/it]
```

```
[ ]: rounded_arrays_2 = [np.round(arr, 3) for arr in prob_array_2]
```

v3

```
[ ]: output_model_path ="./model_path/results_3"
```

```
[ ]: bnb_config = BitsAndBytesConfig(        # 4-bit quantization
             load_in_4bit=True,
             bnb_4bit_quant_type="nf4",
             bnb_4bit_compute_dtype=torch.bfloat16,
             bnb_4bit_use_double_quant=True,
         )

     # To prevent GPU memory overflow in Mixtral8x7b
     config = AutoConfig.from_pretrained('rtzr/ko-gemma-2-9b-it')
     config.gradient_checkpointing = True

     torch.cuda.empty_cache()
     gc.collect()

     model = AutoModelForCausalLM.from_pretrained(
         "rtzr/ko-gemma-2-9b-it",
         device_map="auto",
         quantization_config = bnb_config,
         config=config
     )

     tokenizer = AutoTokenizer.from_pretrained("rtzr/ko-gemma-2-9b-it")
     tokenizer.pad_token = tokenizer.eos_token

     gc.collect()

     # LoRA
     model = PeftModel.from_pretrained(model, output_model_path)
     model.eval()
     print("model loaded")
```

```
config.json:   0%|          | 0.00/852 [00:00<?, ?B/s]

model.safetensors.index.json:   0%|          | 0.00/39.1k [00:00<?, ?B/s]

Downloading shards:   0%|          | 0/10 [00:00<?, ?it/s]

model-00001-of-00010.safetensors:   0%|          | 0.00/1.92G [00:00<?, ?B/s]

model-00002-of-00010.safetensors:   0%|          | 0.00/1.98G [00:00<?, ?B/s]

model-00003-of-00010.safetensors:   0%|          | 0.00/1.98G [00:00<?, ?B/s]

model-00004-of-00010.safetensors:   0%|          | 0.00/1.98G [00:00<?, ?B/s]

model-00005-of-00010.safetensors:   0%|          | 0.00/1.98G [00:00<?, ?B/s]

model-00006-of-00010.safetensors:   0%|          | 0.00/1.98G [00:00<?, ?B/s]

model-00007-of-00010.safetensors:   0%|          | 0.00/1.98G [00:00<?, ?B/s]

model-00008-of-00010.safetensors:   0%|          | 0.00/1.98G [00:00<?, ?B/s]

model-00009-of-00010.safetensors:   0%|          | 0.00/1.98G [00:00<?, ?B/s]

model-00010-of-00010.safetensors:   0%|          | 0.00/705M [00:00<?, ?B/s]
```

```
Loading checkpoint shards:   0%|          | 0/10 [00:00<?, ?it/s]

generation_config.json:   0%|          | 0.00/168 [00:00<?, ?B/s]

tokenizer_config.json:   0%|          | 0.00/40.5k [00:00<?, ?B/s]

tokenizer.model:   0%|          | 0.00/4.24M [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/17.5M [00:00<?, ?B/s]

special_tokens_map.json:   0%|          | 0.00/636 [00:00<?, ?B/s]

model loaded
```

[ ]:

```
##   chat prompt ##
class CustomDataset(Dataset):
    def __init__(self, fname, tokenizer):
        IGNORE_INDEX=-100
        self.inp = []
        self.trg = []
        self.label = []

        PROMPT = '''You are a helpful AI assistant. Please answer the user's␣
 ↪questions kindly.      AI       .                .'''
        answer_dict = {
            "": None,
            "inference_1": 0,
            "inference_2": 1,
            "inference_3": 2
        }

        with open(fname, "r") as f:
            data = json.load(f)

        def make_chat(inp):
            chat = ["[Conversation]"]
            for cvt in inp['conversation']:
                speaker = cvt['speaker']
                utterance = cvt['utterance']
                utterance_id = cvt['utterance_id']
                chat.append(f" {speaker} [{utterance_id}]: {utterance}")
            chat = "\n".join(chat)

            utterance_array = []
            for utterance_id in inp['reference_id']:
                utterance_array.append(utterance_id)  #conversation␣
 ↪utterance_id

            question = f"[Question]\n    {utterance_array}    {inp['category']}"
            if (ord(inp['category'][-1]) - ord(" ")) % 28 > 0:
```

```
            question += " "
        else:
            question += " "

        question += "     ?"

        chat = chat + "\n\n" + question + "\n\n[Option]\n"
        chat += f"A. {inp['inference_1']}\n"
        chat += f"B. {inp['inference_2']}\n"
        chat += f"C. {inp['inference_3']}"

        return chat

    for example in data:
        chat = make_chat(example["input"])
        message = [
            {"role": "system", "content": PROMPT},
            {"role": "user", "content": chat},
        ]
        source = tokenizer.apply_chat_template(
            message,
            add_generation_prompt=True,
            return_tensors="pt",
        )

        target = ""
        if example["output"] == "inference_1":
            target = f"A. {example['input']['inference_1']}{tokenizer.
↪eos_token}"
        elif example["output"] == "inference_2":
            target = f"B. {example['input']['inference_2']}{tokenizer.
↪eos_token}"
        elif example["output"] == "inference_3":
            target = f"C. {example['input']['inference_3']}{tokenizer.
↪eos_token}"


        target = tokenizer(target,
                  return_attention_mask=False,
                  add_special_tokens=False,
                  return_tensors="pt")
        target["input_ids"] = target["input_ids"].type(torch.int64)

        input_ids = torch.concat((source[0], target["input_ids"][0]))
        labels = torch.concat((torch.LongTensor([IGNORE_INDEX] * source[0].
↪shape[0]), target["input_ids"][0]))
```

```
            self.inp.append(input_ids)
            self.label.append(labels)
            self.trg.append(answer_dict[example["output"]])

    def __len__(self):
        return len(self.inp)

    def __getitem__(self, idx):
        return self.inp[idx], self.trg[idx]
```

[ ]:

```
import tqdm
import numpy

dataset = CustomDataset("/dataset/        _   /   _test.json", tokenizer)

answer_dict = {
    0: "inference_1",
    1: "inference_2",
    2: "inference_3",
}

with open("./dataset/        _   /   _test.json", "r") as f:
    result_3 = json.load(f)

with torch.no_grad():
    prob_array_3 = []
    for idx in tqdm.tqdm(range(len(dataset))):
        inp, _ = dataset[idx]
        outputs = model(
            inp.to("cuda:0").unsqueeze(0)
        )
        logits = outputs.logits[:,-1].flatten()
        probs = (
            torch.nn.functional.softmax(
                torch.tensor(
                    [
                        logits[tokenizer.vocab['A']],
                        logits[tokenizer.vocab['B']],
                        logits[tokenizer.vocab['C']],
                    ]
                ),
                dim=0,
            )
            .detach()
            .cpu()
            .to(torch.float32)
            .numpy()
        )

        #     (    )
        prob_array_3.append(probs)

        torch.cuda.empty_cache()
        gc.collect()
```

```
100%|      | 605/605 [13:12<00:00,  1.31s/it]
```

[ ]:
```
rounded_arrays_3 = [np.round(arr, 3) for arr in prob_array_3]
```

v4

[ ]:
```
output_model_path ="./model_path/results_4"
```

[ ]:
```
bnb_config = BitsAndBytesConfig(        # 4-bit quantization
        load_in_4bit=True,
        bnb_4bit_quant_type="nf4",
        bnb_4bit_compute_dtype=torch.bfloat16,
        bnb_4bit_use_double_quant=True,
    )

# To prevent GPU memory overflow in Mixtral8x7b
config = AutoConfig.from_pretrained('chihoonlee10/T3Q-ko-solar-dpo-v7.0')
config.gradient_checkpointing = True

torch.cuda.empty_cache()
gc.collect()

model = AutoModelForCausalLM.from_pretrained(
    "chihoonlee10/T3Q-ko-solar-dpo-v7.0",
    device_map="auto",
    quantization_config = bnb_config,
    config=config
)

tokenizer = AutoTokenizer.from_pretrained("chihoonlee10/T3Q-ko-solar-dpo-v7.0")
tokenizer.pad_token = tokenizer.eos_token

gc.collect()

# LoRA
model = PeftModel.from_pretrained(model, output_model_path)
model.eval()
print("model loaded")
```

```
Loading checkpoint shards:   0%|          | 0/5 [00:00<?, ?it/s]
```

```
model loaded
```

[ ]:

```python
class CustomDataset(Dataset):
    def __init__(self, fname, tokenizer):
        IGNORE_INDEX=-100
        self.inp = []
        self.trg = []
        self.label = []

        PROMPT = '''You are a helpful AI assistant. Please answer the user's␣
 ↪questions correctly.      AI      .               .
                                      :
        1.         .
        2.          .
        3.     (A, B, C)              .
        4.                .'''
        answer_dict = {
            "": None,
            "inference_1": 0,
            "inference_2": 1,
            "inference_3": 2
        }

        with open(fname, "r") as f:
            data = json.load(f)

        def make_add_info(inp):
            question =''

            if  inp['category'] == ' ':
                question += '(            )'
            elif inp['category'] == '   ':
                question += '(              )'
            elif inp['category'] == '   ':
                question += '(                  )'
            elif inp['category'] == '   ':
                question += '(        ‘ ’        )'
            elif inp['category'] == '   ':
                question += '(         ‘ ’          )'

            return question

        def make_chat(inp):
            chat = ["[Conversation]"]
            for cvt in inp['conversation']:
                speaker = cvt['speaker']
                utterance = cvt['utterance']
                chat.append(f" {speaker}: {utterance}   {speaker}  ")
            chat = "\n".join(chat)

            question = f"[Question]\n    {inp['category']}"
            #
            question += make_add_info(inp)

            if (ord(inp['category'][-1]) - ord(" ")) % 28 > 0:
                question += "  "
            else:
                question += " "

            question += "     ?"

            chat = chat + "\n\n" + question + "\n\n[Option]\n"
            chat += f"A. {inp['inference_1']}\n"
            chat += f"B. {inp['inference_2']}\n"
            chat += f"C. {inp['inference_3']}"

            return chat

        for example in data:
            chat = make_chat(example["input"])
            message = [
                {"role": "system", "content": PROMPT},
                {"role": "user", "content": chat},
            ]
            source = tokenizer.apply_chat_template(
                message,
                add_generation_prompt=True,
                return_tensors="pt",
            )

            target = ""
            if example["output"] == "inference_1":
                target = f"A. {example['input']['inference_1']}{tokenizer.eos_token}"
            elif example["output"] == "inference_2":
                target = f"B. {example['input']['inference_2']}{tokenizer.eos_token}"
            elif example["output"] == "inference_3":
                target = f"C. {example['input']['inference_3']}{tokenizer.eos_token}"

            target = tokenizer(target,
                      return_attention_mask=False,
                      add_special_tokens=False,
                      return_tensors="pt")
            target["input_ids"] = target["input_ids"].type(torch.int64)

            input_ids = torch.concat((source[0], target["input_ids"][0]))
            labels = torch.concat((torch.LongTensor([IGNORE_INDEX] * source[0].
↪shape[0]), target["input_ids"][0]))

            self.inp.append(input_ids)
            self.label.append(labels)
            self.trg.append(answer_dict[example["output"]])

    def __len__(self):
        return len(self.inp)

    def __getitem__(self, idx):
        return self.inp[idx], self.trg[idx]
```

[ ]:

```
import tqdm
import numpy

dataset = CustomDataset("./dataset/        _   /   _test.json", tokenizer)

answer_dict = {
    0: "inference_1",
    1: "inference_2",
    2: "inference_3",
}

with open("./dataset/        _   /   _test.json", "r") as f:
    result_4 = json.load(f)

with torch.no_grad():
    prob_array_4 = []
    for idx in tqdm.tqdm(range(len(dataset))):
        inp, _ = dataset[idx]
        outputs = model(
            inp.to("cuda:0").unsqueeze(0)
        )
        logits = outputs.logits[:,-1].flatten()
        probs = (
            torch.nn.functional.softmax(
                torch.tensor(
                    [
                        logits[tokenizer.vocab['A']],
                        logits[tokenizer.vocab['B']],
                        logits[tokenizer.vocab['C']],
                    ]
                ),
                dim=0,
            )
            .detach()
            .cpu()
            .to(torch.float32)
            .numpy()
        )

        #     (    )
        prob_array_4.append(probs)

        torch.cuda.empty_cache()
        gc.collect()
```

```
100%|     | 605/605 [15:29<00:00,  1.54s/it]
```

[ ]:

```
rounded_arrays_4 = [np.round(arr, 3) for arr in prob_array_4]
```

csv

[ ]:

```
import csv
csv_file = './rounded_arrays.csv'

with open(csv_file, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(rounded_arrays)
```

[ ]:

```
csv_file = './rounded_arrays_2.csv'

with open(csv_file, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(rounded_arrays_2)
```

[ ]:

```
csv_file = './rounded_arrays_3.csv'

with open(csv_file, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(rounded_arrays_3)
```

[ ]:

```
csv_file = './rounded_arrays_4.csv'

with open(csv_file, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(rounded_arrays_4)
```

```
[ ]: import pandas as pd

     # CSV        (        header=None  )
     df1 = pd.read_csv('./rounded_arrays.csv', header=None)
     df2 = pd.read_csv('./rounded_arrays_2.csv', header=None)
     df3 = pd.read_csv('./rounded_arrays_3.csv', header=None)
     df4 = pd.read_csv('./rounded_arrays_4.csv', header=None)

     #  DataFrame
     # result_df = (df1 * 0.25) + (df2 * 0.25) + (df3 * 0.25) + (df4 * 0.25) #      ␣
      ↪96.0
     result_df = (df1 * 0.25) + (df2 * 0.30) + (df3 * 0.30) + (df4 * 0.15)

     #      CSV
     result_df.to_csv('./ensemble_result.csv', header=False, index=False)

     print("CSV           .")
```

```
CSV           .
```

```
[ ]: import tqdm
     import numpy

     dataset = CustomDataset("./dataset/        _   /   _test.json", tokenizer)

     answer_dict = {
         0: "inference_1",
         1: "inference_2",
         2: "inference_3",
     }

     with open("./dataset/        _   /   _test.json", "r") as f:
         result_t = json.load(f)

     #
     probs_ensemble = result_df.values.tolist()

     for idx in tqdm.tqdm(range(len(dataset))):

         #
         probs = probs_ensemble[idx]

         result_t[idx]["output"] = answer_dict[numpy.argmax(probs)]


     with open(output, "w", encoding="utf-8") as f:
         f.write(json.dumps(result_t, ensure_ascii=False, indent=4))
```

```
100%|     | 605/605 [00:00<00:00, 138983.13it/s]
```

[ ]:

```

```