Installation and Load packages

```
!pip install datasets peft -qq
!pip install accelerate -qq
!pip install bitsandbytes -qq
!pip install trl -qq

!pip install torch==2.2.0 torchvision==0.17.0 torchaudio==2.2.0 --
index-url https://download.pytorch.org/whl/cu118
!pip install --upgrade --pre transformers accelerate --extra-index-url
https://download.pytorch.org/whl/cu118
!pip install bitsandbytes==0.43.2 --prefer-binary --extra-index-url
https://pypi.org/simple
```

```
Defaulting to user installation because normal site-packages is not
writeable
Looking in indexes: https://download.pytorch.org/whl/cu118
Requirement already satisfied: torch==2.2.0 in
/home/student/.local/lib/python3.10/site-packages (2.2.0+cu118)
Requirement already satisfied: torchvision==0.17.0 in
/home/student/.local/lib/python3.10/site-packages (0.17.0+cu118)
Requirement already satisfied: torchaudio==2.2.0 in
/home/student/.local/lib/python3.10/site-packages (2.2.0+cu118)
Requirement already satisfied: nvidia-cuda-runtime-cu11==11.8.89 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(11.8.89)
Requirement already satisfied: fsspec in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(2024.2.0)
Requirement already satisfied: nvidia-cufft-cu11==10.9.0.58 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(10.9.0.58)
Requirement already satisfied: triton==2.2.0 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(2.2.0)
Requirement already satisfied: nvidia-cusparse-cu11==11.7.5.86 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(11.7.5.86)
Requirement already satisfied: nvidia-nccl-cu11==2.19.3 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(2.19.3)
Requirement already satisfied: nvidia-cuda-nvrtc-cu11==11.8.89 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(11.8.89)
Requirement already satisfied: nvidia-cuda-cupti-cu11==11.8.87 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(11.8.87)
Requirement already satisfied: jinja2 in
```

```
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(3.1.3)
Requirement already satisfied: nvidia-nvtx-cu11==11.8.86 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(11.8.86)
Requirement already satisfied: networkx in
/opt/conda/lib/python3.10/site-packages (from torch==2.2.0) (3.1)
Requirement already satisfied: filelock in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(3.13.1)
Requirement already satisfied: nvidia-curand-cu11==10.3.0.86 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(10.3.0.86)
Requirement already satisfied: sympy in
/opt/conda/lib/python3.10/site-packages (from torch==2.2.0) (1.12)
Requirement already satisfied: nvidia-cudnn-cu11==8.7.0.84 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(8.7.0.84)
Requirement already satisfied: typing-extensions>=4.8.0 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(4.10.0)
Requirement already satisfied: nvidia-cusolver-cu11==11.4.1.48 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(11.4.1.48)
Requirement already satisfied: nvidia-cublas-cu11==11.11.3.6 in
/home/student/.local/lib/python3.10/site-packages (from torch==2.2.0)
(11.11.3.6)
Requirement already satisfied: numpy in
/home/student/.local/lib/python3.10/site-packages (from
torchvision==0.17.0) (1.26.4)
Requirement already satisfied: requests in
/home/student/.local/lib/python3.10/site-packages (from
torchvision==0.17.0) (2.32.3)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in
/home/student/.local/lib/python3.10/site-packages (from
torchvision==0.17.0) (10.2.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/home/student/.local/lib/python3.10/site-packages (from jinja2-
>torch==2.2.0) (2.1.5)
Requirement already satisfied: idna<4,>=2.5 in
/home/student/.local/lib/python3.10/site-packages (from requests-
>torchvision==0.17.0) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/home/student/.local/lib/python3.10/site-packages (from requests-
>torchvision==0.17.0) (2.2.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/home/student/.local/lib/python3.10/site-packages (from requests-
>torchvision==0.17.0) (3.3.2)
Requirement already satisfied: certifi>=2017.4.17 in
```

```
/home/student/.local/lib/python3.10/site-packages (from requests-
>torchvision==0.17.0) (2024.2.2)
Requirement already satisfied: mpmath>=0.19 in
/opt/conda/lib/python3.10/site-packages (from sympy->torch==2.2.0)
(1.3.0)
Defaulting to user installation because normal site-packages is not
writeable
Looking in indexes: https://pypi.org/simple,
https://download.pytorch.org/whl/cu118
Requirement already satisfied: transformers in
/home/student/.local/lib/python3.10/site-packages (4.51.2)
Requirement already satisfied: accelerate in
/home/student/.local/lib/python3.10/site-packages (1.6.0)
Requirement already satisfied: packaging>=20.0 in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(24.0)
Requirement already satisfied: numpy>=1.17 in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(1.26.4)
Requirement already satisfied: requests in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(2.32.3)
Requirement already satisfied: tqdm>=4.27 in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(4.67.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.30.0 in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(0.30.2)
Requirement already satisfied: filelock in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(3.13.1)
Requirement already satisfied: tokenizers<0.22,>=0.21 in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(0.21.1)
Requirement already satisfied: pyyaml>=5.1 in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(6.0.1)
Requirement already satisfied: safetensors>=0.4.3 in
/home/student/.local/lib/python3.10/site-packages (from transformers)
(0.5.3)
Requirement already satisfied: regex!=2019.12.17 in
/opt/conda/lib/python3.10/site-packages (from transformers)
(2023.12.25)
Requirement already satisfied: torch>=2.0.0 in
/home/student/.local/lib/python3.10/site-packages (from accelerate)
(2.2.0+cu118)
Requirement already satisfied: psutil in
/opt/conda/lib/python3.10/site-packages (from accelerate) (5.9.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
```

```
/home/student/.local/lib/python3.10/site-packages (from huggingface-
hub<1.0,>=0.30.0->transformers) (4.10.0)
Requirement already satisfied: fsspec>=2023.5.0 in
/home/student/.local/lib/python3.10/site-packages (from huggingface-
hub<1.0,>=0.30.0->transformers) (2024.2.0)
Requirement already satisfied: nvidia-cusparse-cu11==11.7.5.86 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (11.7.5.86)
Requirement already satisfied: networkx in
/opt/conda/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (3.1)
Requirement already satisfied: nvidia-cufft-cu11==10.9.0.58 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (10.9.0.58)
Requirement already satisfied: nvidia-cudnn-cu11==8.7.0.84 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (8.7.0.84)
Requirement already satisfied: triton==2.2.0 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (2.2.0)
Requirement already satisfied: jinja2 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (3.1.3)
Requirement already satisfied: nvidia-cublas-cu11==11.11.3.6 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (11.11.3.6)
Requirement already satisfied: nvidia-cusolver-cu11==11.4.1.48 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (11.4.1.48)
Requirement already satisfied: nvidia-nccl-cu11==2.19.3 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (2.19.3)
Requirement already satisfied: nvidia-cuda-runtime-cu11==11.8.89 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (11.8.89)
Requirement already satisfied: nvidia-nvtx-cu11==11.8.86 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (11.8.86)
Requirement already satisfied: sympy in
/opt/conda/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (1.12)
Requirement already satisfied: nvidia-cuda-nvrtc-cu11==11.8.89 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (11.8.89)
Requirement already satisfied: nvidia-curand-cu11==10.3.0.86 in
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (10.3.0.86)
Requirement already satisfied: nvidia-cuda-cupti-cu11==11.8.87 in
```

```
/home/student/.local/lib/python3.10/site-packages (from torch>=2.0.0-
>accelerate) (11.8.87)

Requirement already satisfied: charset-normalizer<4,>=2 in
/home/student/.local/lib/python3.10/site-packages (from requests-
>transformers) (3.3.2)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/home/student/.local/lib/python3.10/site-packages (from requests-
>transformers) (2.2.1)
Requirement already satisfied: certifi>=2017.4.17 in
/home/student/.local/lib/python3.10/site-packages (from requests-
>transformers) (2024.2.2)
Requirement already satisfied: idna<4,>=2.5 in
/home/student/.local/lib/python3.10/site-packages (from requests-
>transformers) (3.6)
Requirement already satisfied: MarkupSafe>=2.0 in
/home/student/.local/lib/python3.10/site-packages (from jinja2-
>torch>=2.0.0->accelerate) (2.1.5)
Requirement already satisfied: mpmath>=0.19 in
/opt/conda/lib/python3.10/site-packages (from sympy->torch>=2.0.0-
>accelerate) (1.3.0)
Defaulting to user installation because normal site-packages is not
writeable
Looking in indexes: https://pypi.org/simple, https://pypi.org/simple
Requirement already satisfied: bitsandbytes==0.43.2 in
/home/student/.local/lib/python3.10/site-packages (0.43.2)
Requirement already satisfied: numpy in
/home/student/.local/lib/python3.10/site-packages (from
bitsandbytes==0.43.2) (1.26.4)
Requirement already satisfied: torch in
/home/student/.local/lib/python3.10/site-packages (from
bitsandbytes==0.43.2) (2.2.0+cu118)
Requirement already satisfied: filelock in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (3.13.1)
Requirement already satisfied: networkx in
/opt/conda/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (3.1)
Requirement already satisfied: nvidia-cuda-cupti-cu11==11.8.87 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (11.8.87)
Requirement already satisfied: nvidia-cufft-cu11==10.9.0.58 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (10.9.0.58)
Requirement already satisfied: triton==2.2.0 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (2.2.0)
Requirement already satisfied: nvidia-cudnn-cu11==8.7.0.84 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (8.7.0.84)
```

```
Requirement already satisfied: nvidia-nvtx-cu11==11.8.86 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (11.8.86)
Requirement already satisfied: nvidia-cuda-nvrtc-cu11==11.8.89 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (11.8.89)
Requirement already satisfied: typing-extensions>=4.8.0 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (4.10.0)
Requirement already satisfied: sympy in
/opt/conda/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (1.12)
Requirement already satisfied: nvidia-nccl-cu11==2.19.3 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (2.19.3)
Requirement already satisfied: nvidia-cusolver-cu11==11.4.1.48 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (11.4.1.48)
Requirement already satisfied: nvidia-cusparse-cu11==11.7.5.86 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (11.7.5.86)
Requirement already satisfied: fsspec in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (2024.2.0)
Requirement already satisfied: nvidia-cublas-cu11==11.11.3.6 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (11.11.3.6)
Requirement already satisfied: nvidia-curand-cu11==10.3.0.86 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (10.3.0.86)
Requirement already satisfied: nvidia-cuda-runtime-cu11==11.8.89 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (11.8.89)
Requirement already satisfied: jinja2 in
/home/student/.local/lib/python3.10/site-packages (from torch-
>bitsandbytes==0.43.2) (3.1.3)
Requirement already satisfied: MarkupSafe>=2.0 in
/home/student/.local/lib/python3.10/site-packages (from jinja2->torch-
>bitsandbytes==0.43.2) (2.1.5)
Requirement already satisfied: mpmath>=0.19 in
/opt/conda/lib/python3.10/site-packages (from sympy->torch-
>bitsandbytes==0.43.2) (1.3.0)

!pip install wandb scikit-learn

Defaulting to user installation because normal site-packages is not
writeable
Requirement already satisfied: wandb in
/home/student/.local/lib/python3.10/site-packages (0.19.9)
Requirement already satisfied: scikit-learn in
```

```
/home/student/.local/lib/python3.10/site-packages (1.6.1)
Requirement already satisfied: platformdirs in
/opt/conda/lib/python3.10/site-packages (from wandb) (4.2.0)
Requirement already satisfied: setproctitle in
/home/student/.local/lib/python3.10/site-packages (from wandb) (1.3.5)
Requirement already satisfied: gitpython!=3.1.29,>=1.0.0 in
/home/student/.local/lib/python3.10/site-packages (from wandb)
(3.1.44)
Requirement already satisfied: pyyaml in
/home/student/.local/lib/python3.10/site-packages (from wandb) (6.0.1)
Requirement already satisfied: protobuf!=4.21.0,!=5.28.0,<6,>=3.19.0
in /opt/conda/lib/python3.10/site-packages (from wandb) (4.25.3)
Requirement already satisfied: requests<3,>=2.0.0 in
/home/student/.local/lib/python3.10/site-packages (from wandb)
(2.32.3)
Requirement already satisfied: pydantic<3 in
/home/student/.local/lib/python3.10/site-packages (from wandb) (2.6.4)
Requirement already satisfied: docker-pycreds>=0.4.0 in
/home/student/.local/lib/python3.10/site-packages (from wandb) (0.4.0)
Requirement already satisfied: psutil>=5.0.0 in
/opt/conda/lib/python3.10/site-packages (from wandb) (5.9.0)
Requirement already satisfied: setuptools in
/opt/conda/lib/python3.10/site-packages (from wandb) (65.6.3)
Requirement already satisfied: click!=8.0.0,>=7.1 in
/home/student/.local/lib/python3.10/site-packages (from wandb) (8.1.7)
Requirement already satisfied: typing-extensions<5,>=4.4 in
/home/student/.local/lib/python3.10/site-packages (from wandb)
(4.10.0)
Requirement already satisfied: sentry-sdk>=2.0.0 in
/home/student/.local/lib/python3.10/site-packages (from wandb)
(2.25.1)
Requirement already satisfied: scipy>=1.6.0 in
/opt/conda/lib/python3.10/site-packages (from scikit-learn) (1.11.2)
Requirement already satisfied: numpy>=1.19.5 in
/home/student/.local/lib/python3.10/site-packages (from scikit-learn)
(1.26.4)
Requirement already satisfied: threadpoolctl>=3.1.0 in
/home/student/.local/lib/python3.10/site-packages (from scikit-learn)
(3.6.0)
Requirement already satisfied: joblib>=1.2.0 in
/home/student/.local/lib/python3.10/site-packages (from scikit-learn)
(1.4.2)
Requirement already satisfied: six>=1.4.0 in
/home/student/.local/lib/python3.10/site-packages (from docker-
pycreds>=0.4.0->wandb) (1.16.0)
Requirement already satisfied: gitdb<5,>=4.0.1 in
/home/student/.local/lib/python3.10/site-packages (from gitpython!
=3.1.29,>=1.0.0->wandb) (4.0.12)
Requirement already satisfied: pydantic-core==2.16.3 in
```

```
/home/student/.local/lib/python3.10/site-packages (from pydantic<3-
>wandb) (2.16.3)
Requirement already satisfied: annotated-types>=0.4.0 in
/home/student/.local/lib/python3.10/site-packages (from pydantic<3-
>wandb) (0.6.0)
Requirement already satisfied: idna<4,>=2.5 in
/home/student/.local/lib/python3.10/site-packages (from
requests<3,>=2.0.0->wandb) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/home/student/.local/lib/python3.10/site-packages (from
requests<3,>=2.0.0->wandb) (2.2.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/home/student/.local/lib/python3.10/site-packages (from
requests<3,>=2.0.0->wandb) (3.3.2)
Requirement already satisfied: certifi>=2017.4.17 in
/home/student/.local/lib/python3.10/site-packages (from
requests<3,>=2.0.0->wandb) (2024.2.2)
Requirement already satisfied: smmap<6,>=3.0.1 in
/home/student/.local/lib/python3.10/site-packages (from
gitdb<5,>=4.0.1->gitpython!=3.1.29,>=1.0.0->wandb) (5.0.2)
```

GPU - details

```
import torch

print("Torch version:", torch.__version__)
print("CUDA available:", torch.cuda.is_available())

if torch.cuda.is_available():
    print("Device name:", torch.cuda.get_device_name(0))
else:
    print("No GPU detected.")
```

```
Torch version: 2.2.0+cu118
CUDA available: True
Device name: Tesla T4
```

Load libraries, Login HuggingFace API & WandB API

- **HuggingFace API:** To get access of Model Llama-3 (8 Billion)
- **WandB (Weigths & Biases):** To supervise perform of model and hyperparameter Tuning

```
# from google.colab import userdata
from huggingface_hub import login

login(token="YOUR_HF_API_KEY")
```

```
# Access Key for llama Model (HuggingFace)

from datasets import load_dataset, Dataset
from sklearn.model_selection import train_test_split

from transformers import (
    AutoTokenizer,
    AutoModelForCausalLM,
    TrainingArguments,
    DataCollatorForLanguageModeling,
    Trainer,
    BitsAndBytesConfig,
    HfArgumentParser,
    pipeline,
    logging
)

from peft import (
    LoraConfig,
    PeftModel,
    prepare_model_for_kbit_training,
    get_peft_model,
)
from bitsandbytes.optim import AdamW8bit
import os, torch, wandb
from trl import SFTTrainer, setup_chat_format
```

WandB - For plot Training

```
# for hyperparameter tuning report
wandb.login()

wandb.init(project='PH-Llama-3.0',
    job_type="training",)
# YOUR_WANDB_API_KEY
```

```
wandb: WARNING Calling wandb.login() after wandb.init() has no effect.
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<wandb.sdk.wandb_run.Run at 0x7fbb55ed5420>
```

Load Llama-3.2-3B model

```
model_name = "meta-llama/Llama-3.2-3B"

bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.bfloat16,
    bnb_4bit_use_double_quant=False
)


tokenizer = AutoTokenizer.from_pretrained(model_name)

base_model = AutoModelForCausalLM.from_pretrained(
    model_name,
    device_map="auto",
    quantization_config=bnb_config,
     attn_implementation="eager"
)

if tokenizer.pad_token is None:
    tokenizer.pad_token = tokenizer.eos_token

base_model.config.pretraining_tp = 1
base_model.config.use_cache = False

{"model_id":"162e904c400445989e032e70c5996153","version_major":2,"vers
ion_minor":0}

print(f"meta-llama/Llama-3.2-3B:\n\n{base_model}")

meta-llama/Llama-3.2-3B:

LlamaForCausalLM(
  (model): LlamaModel(
    (embed_tokens): Embedding(128256, 3072)
    (layers): ModuleList(
      (0-27): 28 x LlamaDecoderLayer(
        (self_attn): LlamaAttention(
          (q_proj): Linear4bit(in_features=3072, out_features=3072,
```

```
bias=False)
          (k_proj): Linear4bit(in_features=3072, out_features=1024,
bias=False)
          (v_proj): Linear4bit(in_features=3072, out_features=1024,
bias=False)
          (o_proj): Linear4bit(in_features=3072, out_features=3072,
bias=False)
        )
        (mlp): LlamaMLP(
          (gate_proj): Linear4bit(in_features=3072, out_features=8192,
bias=False)
          (up_proj): Linear4bit(in_features=3072, out_features=8192,
bias=False)
          (down_proj): Linear4bit(in_features=8192, out_features=3072,
bias=False)
          (act_fn): SiLU()
        )
        (input_layernorm): LlamaRMSNorm((3072,), eps=1e-05)
        (post_attention_layernorm): LlamaRMSNorm((3072,), eps=1e-05)
      )
    )
    (norm): LlamaRMSNorm((3072,), eps=1e-05)
    (rotary_emb): LlamaRotaryEmbedding()
  )
  (lm_head): Linear(in_features=3072, out_features=128256, bias=False)
)

print(f"{base_model.config}")

LlamaConfig {
  "_attn_implementation_autoset": true,
  "architectures": [
    "LlamaForCausalLM"
  ],
  "attention_bias": false,
  "attention_dropout": 0.0,
  "bos_token_id": 128000,
  "eos_token_id": 128001,
  "head_dim": 128,
  "hidden_act": "silu",
  "hidden_size": 3072,
  "initializer_range": 0.02,
  "intermediate_size": 8192,
  "max_position_embeddings": 131072,
  "mlp_bias": false,
  "model_type": "llama",
  "num_attention_heads": 24,
  "num_hidden_layers": 28,
  "num_key_value_heads": 8,
  "pretraining_tp": 1,
```

```
  "quantization_config": {
    "_load_in_4bit": true,
    "_load_in_8bit": false,
    "bnb_4bit_compute_dtype": "bfloat16",
    "bnb_4bit_quant_storage": "uint8",
    "bnb_4bit_quant_type": "nf4",
    "bnb_4bit_use_double_quant": false,
    "llm_int8_enable_fp32_cpu_offload": false,
    "llm_int8_has_fp16_weight": false,
    "llm_int8_skip_modules": null,
    "llm_int8_threshold": 6.0,
    "load_in_4bit": true,
    "load_in_8bit": false,
    "quant_method": "bitsandbytes"
  },
  "rms_norm_eps": 1e-05,
  "rope_scaling": {
    "factor": 32.0,
    "high_freq_factor": 4.0,
    "low_freq_factor": 1.0,
    "original_max_position_embeddings": 8192,
    "rope_type": "llama3"
  },
  "rope_theta": 500000.0,
  "tie_word_embeddings": true,
  "torch_dtype": "float16",
  "transformers_version": "4.51.2",
  "use_cache": false,
  "vocab_size": 128256
}
```


Trainable parameters - Model

```
def trainable_parameters(model):
    """
    Prints the number of trainable parameters in the model.
    """
    trainable_params = 0
    all_param = 0
    for _, param in model.named_parameters():
        all_param += param.numel()
        if param.requires_grad:
            trainable_params += param.numel()
    return f"- Trainable model parameters: {trainable_params}.\n- All model parameters: {all_param}.\n- Percentage of trainable model parameters: {100 * trainable_params / all_param:.2f}%"

print(trainable_parameters(base_model))
```

```
- Trainable model parameters: 394177536.
- All model parameters: 1803463680.
- Percentage of trainable model parameters: 21.86%
```

Assign datasetPH.json

Data is split in to train and test.

- Train size: 80%
- Test size: 20%

```
import json
with open("./dataset/datasetPH.json", "r") as f:
    data = json.load(f)

if isinstance(data, dict):
    print("Data is a dictionary. Converting values to a list for
splitting.")
    data = list(data.values())

train_data, test_data = train_test_split(data, test_size=0.2,
random_state=42)

with open("./dataset/train_datasetPH.json", "w") as f:
    json.dump(train_data, f, indent=2)

with open("./dataset/test_datasetPH.json", "w") as f:
    json.dump(test_data, f, indent=2)

print(f"Train size: {len(train_data)}")
print(f"Test size: {len(test_data)}")
```

```
Data is a dictionary. Converting values to a list for splitting.
Train size: 160
Test size: 41
```

```
data[0]
```

```
{'paper_id': 'ED012836',
 'title': 'Adult Basic Education Work Book in Basic Arithmetic. Parts
I and II.',
 'author': 'Graham, Minnie M.',
 'publication_year': 1966,
 'source': 'Danbury Public Schools, Connecticut',
 'doi_or_url': '',
 'topic_category': 'Adult Education / Arithmetic Instruction',
 'document_type': 'Workbook',
 'abstract': 'These workbooks provide teaching materials and drill
exercises in multiplication for adult basic education learners in
Danbury, Connecticut. Part I covers multiplication by numbers two
through nine, while Part II expands to ten through twelve, including
```

```
dollars and cents, and offers speed and accuracy drills.',
 'key_findings': 'Instructional workbooks tailored for adult learners
can assist in foundational arithmetic, especially multiplication,
through structured drills and exercises.',
 'problem_statement': 'Adult learners require appropriately designed
arithmetic materials to support basic educational needs at elementary
levels.',
 'objectives': 'To provide instructors with multiplication drill
materials suitable for adults operating at elementary grade levels.',
 'conclusion': 'The workbook is a supportive instructional aid that
must be supplemented with additional materials and practice exercises
to effectively meet adult learners' needs.',
 'methodology': {'data_sources': 'Experience and requests from
arithmetic teachers of adult students.',
  'methods_used': 'Instructional material design and exercise
formulation.',
  'sample_size': None,
  'duration': '1966-1967 academic year',
  'research_design': 'Development and application of a structured
workbook for classroom use.'},
 'metrics_and_indicators': [{'metric_name': 'Speed and accuracy in
multiplication',
   'metric_value': None}],
 'policy_practice_implications': {'recommendations': 'Use the workbook
as a teaching aid for adult learners needing arithmetic instruction at
elementary level.',
  'implementation_notes': 'Instructors should supplement with
additional materials to ensure comprehensive understanding.'},
 'thematic_dimensions': {'demographic_focus': 'Adults in basic
education programs',
  'geographic_scope': 'Danbury, Connecticut',
  'domain_keywords': ['multiplication',
   'adult education',
   'arithmetic',
   'instructional materials',
   'workbook']},
 'comparative_and_qualitative_insights': {'comparative_data': '',
  'thematic_analysis': 'Focuses on gradual progression from simpler to
more complex multiplication problems with contextual financial
applications.',
  'limitations': 'Workbook alone is insufficient for comprehensive
instruction.',
  'future_work': 'Create additional supporting materials and broader
coverage of arithmetic topics.'},
 'supporting_materials': {'tables': ['Multiplication tables from 2
through 12'],
  'charts': [],
  'appendices': [],
```

```
  'external_links': []},
 'references': []}
```

Tokenization of dataset and normalization

```
# def tokenize_function(examples):
#     texts = []
#     for i in range(len(examples["title"])):
#         entry_parts = []

#         for key in examples.keys():
#             value = examples[key][i]
#             if isinstance(value, dict):
#                 for subkey, subval in value.items():
#                     entry_parts.append(f"{key}.{subkey}: {subval}")
#             elif isinstance(value, list):
#                 entry_parts.append(f"{key}: {', '.join(map(str,
value))}")
#             else:
#                 entry_parts.append(f"{key}: {value}")

#         combined_text = "\n".join(entry_parts)
#         texts.append(combined_text)

#     return tokenizer(texts, truncation=True, padding="max_length",
max_length=256)


def tokenize_function(examples):
    prompts = []
    for i in range(len(examples["title"])):
        entry = {key: examples[key][i] for key in examples}
        full_prompt = build_prompt(entry)
        prompts.append(full_prompt)

    return tokenizer(prompts, truncation=True, padding="max_length",
max_length=512)

def normalize_entry(entry):
    normalized = {}
    for key, value in entry.items():
        if isinstance(value, dict):
            for subkey, subval in value.items():
                normalized[f"{key}.{subkey}"] = str(subval) if subval
is not None else ""
        elif isinstance(value, list):
            normalized[key] = ", ".join(map(str, value))
        elif value is None:
            normalized[key] = ""
        else:
```

```
            normalized[key] = str(value)
    return normalized

# Normalize each entry
train_data_clean = [normalize_entry(entry) for entry in train_data]
test_data_clean = [normalize_entry(entry) for entry in test_data]

train_dataset_hf = Dataset.from_list(train_data_clean)
test_dataset_hf = Dataset.from_list(test_data_clean)
```

Prompt Engineering

```
def build_prompt(entry):
    # Define the analyst's persona with added expertise details
    persona = (
        "You are an expert public policy analyst specializing in
educational reform and adult education. "
        "Your expertise includes evaluating instructional materials
and their impact on adult learning.\n"
    )

    # Provide clear and detailed instructions including expected
structure and additional considerations
    instruction = (
        "Your task is to analyze the report provided below and
summarize its key findings. "
        "Your output must include:\n"
        "- Three concise bullet points summarizing the findings\n"
        "- One well-structured paragraph discussing the implications,
including any potential policy recommendations or risks\n"
        "- A JSON object tagged with `impact` (possible values:
positive, negative, or neutral) based on the report's overall impact\
n"
    )

    # Add a metadata section with relevant background details
    metadata = (
        f"Metadata:\n"
        f"Paper ID: {entry.get('paper_id', '')}\n"
        f"Title: {entry.get('title', '')}\n"
        f"Author: {entry.get('author', '')}\n"
        f"Publication Year: {entry.get('publication_year', '')}\n"
        f"Source: {entry.get('source', '')}\n"
        f"Document Type: {entry.get('document_type', '')}\n"
        f"Topic Category: {entry.get('topic_category', '')}\n\n"
    )

    # Provide contextual background using details from the entry and
emphasizing audience and local context
```

```
    context = (
        f"This report evaluates an adult education intervention
designed to improve arithmetic skills through instructional workbooks.
"
        f"The intervention was implemented in
{entry.get('thematic_dimensions', {}).get('geographic_scope', 'a
specific region')} and primarily targets
{entry.get('thematic_dimensions', {}).get('demographic_focus', 'adult
learners')}.\n"
    )

    format_guide = (
        "Use a professional and analytical tone with clarity and
conciseness. "
        "Structure your response with bullet points, followed by a
paragraph, and then a JSON object.\n"
    )

    few_shot = (
        "Example Input: \"The policy resulted in 70% improvement in
adult math scores and significantly lowered dropout rates.\"\n"
        "Example Output:\n"
        "- Improved math proficiency by 70%\n"
        "- Significantly reduced dropout rates\n"
        "- Increased learner engagement\n"
        "Implication: The results indicate that the program is
effective and scalable, suggesting positive future impacts on adult
education.\n"
        "{\"impact\": \"positive\"}\n"
    )

    # Construct the body of the report by concisely combining key
parts of the report
    full_text = (
        f"Abstract: {entry.get('abstract', '')}\n"
        f"Key Findings: {entry.get('key_findings', '')}\n"
        f"Problem Statement: {entry.get('problem_statement', '')}\n"
        f"Objectives: {entry.get('objectives', '')}\n"
        f"Conclusion: {entry.get('conclusion', '')}\n"
        f"Methodology: {entry.get('methodology',
{}).get('methods_used', '')}, based on data from
{entry.get('methodology', {}).get('data_sources', '')}, conducted over
{entry.get('methodology', {}).get('duration', '')}\n"
        f"Implications: {entry.get('policy_practice_implications',
{}).get('recommendations', '')}
{entry.get('policy_practice_implications',
{}).get('implementation_notes', '')}\n"
        f"Thematic Focus: {entry.get('thematic_dimensions',
{}).get('demographic_focus', '')} | {entry.get('topic_category', '')}\
```

```
n"
        f"Limitations:
{entry.get('comparative_and_qualitative_insights',
{}).get('limitations', '')}\n"
        f"Future Work:
{entry.get('comparative_and_qualitative_insights',
{}).get('future_work', '')}\n"
    )

    return persona + instruction + metadata + context + format_guide +
few_shot + "Now analyze this report:\n" + full_text
```

Train & Test - Tokenization

```
tokenized_train = train_dataset_hf.map(tokenize_function,
batched=True)
tokenized_train.set_format(type="torch")
print("Tokenization complete with all features.")
```


```
{"model_id":"c20063269c104b6197a9618522ca336e","version_major":2,"vers
ion_minor":0}
```


```
Tokenization complete with all features.
```

```
tokenized_test = test_dataset_hf.map(tokenize_function, batched=True)
tokenized_test.set_format(type="torch")
print("Tokenization complete with all features.")
```


```
{"model_id":"3e37b4e3178b4ccaa91eb1caddfaad95","version_major":2,"vers
ion_minor":0}
```


```
Tokenization complete with all features.
```

Configer - PEFT, LoRA & QLoRA

```
lora_config = LoraConfig(
    r=16,
    lora_alpha=32,
#     target_modules=["q_proj", "v_proj"],
    target_modules=['up_proj', 'down_proj', 'gate_proj', 'k_proj',
'q_proj', 'v_proj', 'o_proj'],
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM"
)

base_model.gradient_checkpointing_enable()
base_model = prepare_model_for_kbit_training(base_model)
```

```
peft_model = get_peft_model(base_model, lora_config)
peft_model.config.use_cache = False

print("After PEFT wrapping:")
print(trainable_parameters(peft_model))
```

```
After PEFT wrapping:
- Trainable model parameters: 24313856.
- All model parameters: 1827777536.
- Percentage of trainable model parameters: 1.33%
```

Train PH-Llama-3.2 Model & Evaluation

```
import torch
import os
data_collator = DataCollatorForLanguageModeling(tokenizer=tokenizer,
mlm=False)

os.environ["PYTORCH_CUDA_ALLOC_CONF"] = "expandable_segments:True"

training_args = TrainingArguments(
    output_dir="./PH-Llama-3.1",
    overwrite_output_dir=True,
    per_device_train_batch_size=1,
    per_device_eval_batch_size=1,
    gradient_accumulation_steps=2,
    optim="paged_adamw_32bit",
    num_train_epochs=5,
    eval_strategy="steps",
    eval_steps=50,
    logging_steps=1,
    weight_decay=0.01,
    warmup_steps=10,
    logging_strategy="steps",
    learning_rate=2e-4,
    fp16=True,
    bf16=False,
    group_by_length=True,
    report_to="wandb"

#     num_train_epochs=5,
#     per_device_train_batch_size=1,
#     per_device_eval_batch_size=1,
#     gradient_accumulation_steps=1,
#     learning_rate=2e-5,
#     weight_decay=0.01,
#     logging_steps=10,
#     save_steps=100,
```

```
#     eval_strategy="steps",
#     eval_steps=50,
#     save_total_limit=2,
#     fp16=True,
#     report_to="wandb"
)

trainer = SFTTrainer(
    model=peft_model,
    args=training_args,
    peft_config=lora_config,
    train_dataset=tokenized_train,
    eval_dataset=tokenized_test,
    data_collator=data_collator,
    optimizers=(AdamW8bit(peft_model.parameters(), lr=2e-5), None)
)

torch.cuda.empty_cache() # Force Clear Cache Before Training

print("Starting training...")
trainer.train()
print("Training complete.")
```


```
{"model_id":"c3fe45f1fe374e189a348caca5f31399","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"25aa3c74a53d4949a78b2956b5326c4f","version_major":2,"vers
ion_minor":0}
```


```
No label_names provided for model class `PeftModelForCausalLM`. Since
`PeftModel` hides base models input arguments, if label_names is not
given, label_names can't be set automatically within `Trainer`. Note
that empty label_names list will be used instead.
wandb: WARNING The `run_name` is currently set to the same value as
`TrainingArguments.output_dir`. If this was not intended, please
specify a different run name by setting the
`TrainingArguments.run_name` parameter.

Starting training...

wandb: Using wandb-core as the SDK backend.  Please refer to
https://wandb.me/wandb-core for more information.
wandb: Currently logged in as: yashnayi00 (yashnayi00-university-of-
new-haven) to https://api.wandb.ai. Use `wandb login --relogin` to
force relogin
/home/student/.local/lib/python3.10/site-packages/pydantic/main.py:314
: UserWarning: Pydantic serializer warnings:
  Expected `list[str]` but got `tuple` - serialized value may not be
as expected
  Expected `list[str]` but got `tuple` - serialized value may not be
```

```
as expected
  return self.__pydantic_serializer__.to_python(

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

Training complete.
```

```
eval_results = trainer.evaluate()
print("Evaluation Results:")
print(eval_results)
```

```
Evaluation Results:
{'eval_loss': 0.7654125690460205, 'eval_runtime': 20.84,
'eval_samples_per_second': 1.967, 'eval_steps_per_second': 1.967}
```

```
peft_model.config.save_pretrained("./PH-Llama-3.1")
```

```
!ls -la ./PH-Llama-3.1
```

```
huggingface/tokenizers: The current process just got forked, after
parallelism has already been used. Disabling parallelism to avoid
deadlocks...
To disable this warning, you can either:
     - Avoid using `tokenizers` before the fork if possible
     - Explicitly set the environment variable
TOKENIZERS_PARALLELISM=(true | false)

total 16
drwxr-xr-x 3 student student 4096 Apr 11 17:27 .
drwxr-xr-x 7 student student 4096 Apr 11 17:27 ..
drwxr-xr-x 2 student student 4096 Apr 11 17:23 checkpoint-400
-rw-r--r-- 1 student student 1361 Apr 11 17:27 config.json
```

```
files = os.listdir("./PH-Llama-3.1")
print("Files in the output directory:", files)
```

```
Files in the output directory: ['checkpoint-400', 'config.json']
```

Generate Text by Trained Model

```
def generate_text(prompt, max_length=100, temperature=0.7,
top_p=0.95):
    inputs = tokenizer(prompt, return_tensors="pt", padding=True,
truncation=True)
    inputs = {key: value.to(peft_model.device) for key, value in
inputs.items()}

    outputs = peft_model.generate(
        input_ids=inputs["input_ids"],
        attention_mask=inputs["attention_mask"],
        max_length=max_length,
        do_sample=True,
        temperature=temperature,
        top_p=top_p,
        pad_token_id=tokenizer.eos_token_id
    )
    generated_text = tokenizer.decode(outputs[0],
skip_special_tokens=True)
    return generated_text

# prompt = build_prompt("Using the dataset from the Peterson-KFF
Health System Tracker on U.S. healthcare quality, provide a
comprehensive analysis comparing the United States to other high-
income countries. In your response, summarize key metrics such as life
expectancy, all-cause mortality, maternal mortality, and rates of
premature death. Discuss the impact of socioeconomic factors and
healthcare utilization on these outcomes, and explain why the U.S. may
perform worse on several indicators despite high per capita
spending.")
# print(generate_text(prompt, max_length=512))

def build_prompt_gen(entry):
    # If the input isn't a dictionary, wrap it into one with default
values
    if not isinstance(entry, dict):
        entry = {
            "paper_id": "",
            "title": "",
            "author": "",
            "publication_year": "",
            "source": "",
            "document_type": "",
            "topic_category": "",
            "abstract": entry,
            "key_findings": "",
            "problem_statement": "",
            "objectives": "",
            "conclusion": "",
```

```
            "methodology": {"methods_used": "", "data_sources": "",
"duration": ""},
            "policy_practice_implications": {"recommendations": "",
"implementation_notes": ""},
            "thematic_dimensions": {"geographic_scope": "a specific
region", "demographic_focus": ""},
            "comparative_and_qualitative_insights": {"limitations":
"", "future_work": ""}
        }
    persona = (
        "You are an expert public policy analyst specializing in
educational reform and adult education. "
        "Your expertise includes evaluating instructional materials
and their impact on adult learning.\n"
    )

    instruction = (
        "Your task is to analyze the report provided below and
summarize its key findings. "
        "Your output must include:\n"
        "- Three concise bullet points summarizing the findings\n"
        "- One well-structured paragraph discussing the implications,
including any potential policy recommendations or risks\n"
        "- A JSON object tagged with `impact` (possible values:
positive, negative, or neutral) based on the report's overall impact\
n"
    )

    metadata = (
        f"Metadata:\n"
        f"Paper ID: {entry.get('paper_id', '')}\n"
        f"Title: {entry.get('title', '')}\n"
        f"Author: {entry.get('author', '')}\n"
        f"Publication Year: {entry.get('publication_year', '')}\n"
        f"Source: {entry.get('source', '')}\n"
        f"Document Type: {entry.get('document_type', '')}\n"
        f"Topic Category: {entry.get('topic_category', '')}\n\n"
    )

    context = (
        f"This report evaluates an adult education intervention
designed to improve arithmetic skills through instructional workbooks.
"
        f"The intervention was implemented in
{entry.get('thematic_dimensions', {}).get('geographic_scope', 'a
specific region')} and primarily targets
{entry.get('thematic_dimensions', {}).get('demographic_focus', 'adult
learners')}.\n"
    )
```

```
    format_guide = (
        "Use a professional and analytical tone with clarity and
conciseness. "
        "Structure your response with bullet points, followed by a
paragraph, and then a JSON object.\n"
    )

    few_shot = (
        "Example Input: \"The policy resulted in 70% improvement in
adult math scores and significantly lowered dropout rates.\"\n"
        "Example Output:\n"
        "- Improved math proficiency by 70%\n"
        "- Significantly reduced dropout rates\n"
        "- Increased learner engagement\n"
        "Implication: The results indicate that the program is
effective and scalable, suggesting positive future impacts on adult
education.\n"
        "{\"impact\": \"positive\"}\n"
    )

    full_text = (
        f"Abstract: {entry.get('abstract', '')}\n"
        f"Key Findings: {entry.get('key_findings', '')}\n"
        f"Problem Statement: {entry.get('problem_statement', '')}\n"
        f"Objectives: {entry.get('objectives', '')}\n"
        f"Conclusion: {entry.get('conclusion', '')}\n"
        f"Methodology: {entry.get('methodology',
{}).get('methods_used', '')}, based on data from
{entry.get('methodology', {}).get('data_sources', '')}, conducted over
{entry.get('methodology', {}).get('duration', '')}\n"
        f"Implications: {entry.get('policy_practice_implications',
{}).get('recommendations', '')}
{entry.get('policy_practice_implications',
{}).get('implementation_notes', '')}\n"
        f"Thematic Focus: {entry.get('thematic_dimensions',
{}).get('demographic_focus', '')} | {entry.get('topic_category', '')}\
n"
        f"Limitations:
{entry.get('comparative_and_qualitative_insights',
{}).get('limitations', '')}\n"
        f"Future Work:
{entry.get('comparative_and_qualitative_insights',
{}).get('future_work', '')}\n"
    )

    return persona + instruction + metadata + context + format_guide +
few_shot + "Now analyze this report:\n" + full_text
```

```
# Usage
raw_context = "Using the dataset from the Peterson-KFF Health System
Tracker on U.S. healthcare quality, provide a comprehensive analysis
comparing the United States to other high-income countries..."
prompt = build_prompt(raw_context)
print(generate_text(prompt, max_length=512))

---------------------------------------------------------------------------
-----
AttributeError                            Traceback (most recent call
last)
Cell In[58], line 83
     81 # Usage
     82 raw_context = "Using the dataset from the Peterson-KFF Health
System Tracker on U.S. healthcare quality, provide a comprehensive
analysis comparing the United States to other high-income
countries..."
---> 83 prompt = build_prompt(raw_context)
     84 print(generate_text(prompt, max_length=512))

Cell In[38], line 20, in build_prompt(entry)
      9 instruction = (
     10     "Your task is to analyze the report provided below and
summarize its key findings. "
     11     "Your output must include:\n"
   (...)
     14     "- A JSON object tagged with `impact` (possible values:
positive, negative, or neutral) based on the report’s overall impact\
n"
     15 )
     17 # Add a metadata section with relevant background details
     18 metadata = (
     19     f"Metadata:\n"
---> 20     f"Paper ID: {entry.get('paper_id', '')}\n"
     21     f"Title: {entry.get('title', '')}\n"
     22     f"Author: {entry.get('author', '')}\n"
     23     f"Publication Year: {entry.get('publication_year', '')}\n"
     24     f"Source: {entry.get('source', '')}\n"
     25     f"Document Type: {entry.get('document_type', '')}\n"
     26     f"Topic Category: {entry.get('topic_category', '')}\n\n"
     27 )
     29 # Provide contextual background using details from the entry
and emphasizing audience and local context
     30 context = (
     31     f"This report evaluates an adult education intervention
designed to improve arithmetic skills through instructional workbooks.
"
     32     f"The intervention was implemented in
{entry.get('thematic_dimensions', {}).get('geographic_scope', 'a
specific region')} and primarily targets
```

```
{entry.get('thematic_dimensions', {}).get('demographic_focus', 'adult
learners')}.\n"
     33 )

AttributeError: 'str' object has no attribute 'get'

prompt = """U.S. Healthcare vs. Other High-Income Countries abstract
This report compares the quality of healthcare in the United States to
other high-income countries,
focusing on key metrics such as life expectancy, all-cause mortality,
maternal mortality, and premature death.
It discusses how high healthcare spending in the U.S. does not
translate into better outcomes."""

prompt = build_prompt_gen(prompt)
print(generate_text(prompt, max_length=512))

You are an expert public policy analyst specializing in educational
reform and adult education. Your expertise includes evaluating
instructional materials and their impact on adult learning.
Your task is to analyze the report provided below and summarize its
key findings. Your output must include:
- Three concise bullet points summarizing the findings
- One well-structured paragraph discussing the implications, including
any potential policy recommendations or risks
- A JSON object tagged with `impact` (possible values: positive,
negative, or neutral) based on the report's overall impact
Metadata:
Paper ID:
Title:
Author:
Publication Year:
Source:
Document Type:
Topic Category:

This report evaluates an adult education intervention designed to
improve arithmetic skills through instructional workbooks. The
intervention was implemented in a specific region and primarily
targets.
Use a professional and analytical tone with clarity and conciseness.
Structure your response with bullet points, followed by a paragraph,
and then a JSON object.
Example Input: "The policy resulted in 70% improvement in adult math
scores and significantly lowered dropout rates."
Example Output:
- Improved math proficiency by 70%
- Significantly reduced dropout rates
- Increased learner engagement
Implication: The results indicate that the program is effective and
```

```
scalable, suggesting positive future impacts on adult education.
{"impact": "positive"}
Now analyze this report:
Abstract: U.S. Healthcare vs. Other High-Income Countries abstract
This report compares the quality of healthcare in the United States to
other high-income countries,
focusing on key metrics such as life expectancy, all-cause mortality,
maternal mortality, and premature death.
It discusses how high healthcare spending in the U.S. does not
translate into better outcomes.
Key Findings:
Problem Statement:
Objectives:
Conclusion:
Methodology:, based on data from, conducted over
Implications:
Thematic Focus:  |
Limitations:
Future Work:
Policy Implications:
Legal Regulations:
Ethical Considerations:
Personal Bias:
Paper ID: HIP-2021-01
Title: Healthcare in the U.S.: A Comparative Perspective
Author:
Publication Year: 2021
Source: Independent Research Institution
Document Type: Policy Report
Topic Category: Health Policy / International Healthcare

Solution: Your output must include:
: Three concise bullet points
: A well-structured paragraph
: A tagged JSON object
Metadata: Paper ID: HIP-2021-01
Title: Healthcare in the U.S.: A Comparative Perspective

entry_1 = {
    "title": "Comparative Analysis of U.S. Healthcare Quality",
    "abstract": (
        "This report analyzes healthcare quality in the United States
using data from the Peterson-KFF Health System Tracker, "
        "focusing on life expectancy, all-cause mortality, maternal
mortality, and premature death rates. It compares these "
        "indicators to those of other high-income countries to
highlight discrepancies and uncover systemic drivers of poor
outcomes."
    ),
    "key_findings": (
```

```
        "- The U.S. has one of the lowest life expectancies among OECD
nations.\n"
        "- Maternal mortality in the U.S. is more than double that of
the next highest country.\n"
        "- The U.S. leads in rates of avoidable premature deaths
despite high spending."
    ),
    "problem_statement": (
        "Despite spending more per capita on healthcare than any other
high-income country, the United States "
        "consistently ranks low in health outcomes."
    ),
    "objectives": (
        "To investigate why the U.S. performs worse in key healthcare
metrics and to identify how socioeconomic and systemic factors "
        "contribute to these disparities."
    ),
    "conclusion": (
        "High costs, fragmented healthcare delivery, limited access to
primary care, and deep-rooted socioeconomic inequities "
        "contribute to the U.S.’s underperformance. Investment in
social services and system-wide reform is needed."
    ),
    "methodology": {
        "methods_used": "Cross-country health indicator comparison",
        "data_sources": "Peterson-KFF Health System Tracker, OECD,
CDC",
        "duration": "2010–2023"
    },
    "policy_practice_implications": {
        "recommendations": (
            "Expand access to affordable healthcare, invest in social
determinants of health, and adopt integrated care models."
        ),
        "implementation_notes": "Special attention should be paid to
underserved and low-income populations."
    },
    "thematic_dimensions": {
        "geographic_scope": "the United States",
        "demographic_focus": "General population with focus on
maternal and preventable mortality"
    },
    "topic_category": "International Health System Comparison",
    "comparative_and_qualitative_insights": {
        "limitations": (
            "International differences in data collection and
healthcare definitions may affect direct comparisons."
        ),
        "future_work": (
```

```
            "Explore policy interventions from high-performing
countries that can be adapted to the U.S. context."
        )
    }
}

prompt = build_prompt(entry_1)
print(generate_text(prompt, max_length=1024))
```

```
You are an expert public policy analyst specializing in educational
reform and adult education. Your expertise includes evaluating
instructional materials and their impact on adult learning.
Your task is to analyze the report provided below and summarize its
key findings. Your output must include:
- Three concise bullet points summarizing the findings
- One well-structured paragraph discussing the implications, including
any potential policy recommendations or risks
- A JSON object tagged with `impact` (possible values: positive,
negative, or neutral) based on the report's overall impact
Metadata:
Paper ID:
Title: Comparative Analysis of U.S. Healthcare Quality
Author:
Publication Year:
Source:
Document Type:
Topic Category: International Health System Comparison

This report evaluates an adult education intervention designed to
improve arithmetic skills through instructional workbooks. The
intervention was implemented in the United States and primarily
targets General population with focus on maternal and preventable
mortality.
Use a professional and analytical tone with clarity and conciseness.
Structure your response with bullet points, followed by a paragraph,
and then a JSON object.
Example Input: "The policy resulted in 70% improvement in adult math
scores and significantly lowered dropout rates."
Example Output:
- Improved math proficiency by 70%
- Significantly reduced dropout rates
- Increased learner engagement
Implication: The results indicate that the program is effective and
scalable, suggesting positive future impacts on adult education.
{"impact": "positive"}
Now analyze this report:
Abstract: This report analyzes healthcare quality in the United States
using data from the Peterson-KFF Health System Tracker, focusing on
life expectancy, all-cause mortality, maternal mortality, and
premature death rates. It compares these indicators to those of other
```

high-income countries to highlight discrepancies and uncover systemic drivers of poor outcomes.
Key Findings: - The U.S. has one of the lowest life expectancies among OECD nations.
- Maternal mortality in the U.S. is more than double that of the next highest country.
- The U.S. leads in rates of avoidable premature deaths despite high spending.
Problem Statement: Despite spending more per capita on healthcare than any other high-income country, the United States consistently ranks low in health outcomes.
Objectives: To investigate why the U.S. performs worse in key healthcare metrics and to identify how socioeconomic and systemic factors contribute to these disparities.
Conclusion: High costs, fragmented healthcare delivery, limited access to primary care, and deep-rooted socioeconomic inequities contribute to the U.S.’s underperformance. Investment in social services and system-wide reform is needed.
Methodology: Cross-country health indicator comparison, based on data from Peterson-KFF Health System Tracker, OECD, CDC, conducted over 2010–2023
Implications: Expand access to affordable healthcare, invest in social determinants of health, and adopt integrated care models. Special attention should be paid to underserved and low-income populations.
Thematic Focus: General population with focus on maternal and preventable mortality | International Health System Comparison
Limitations: International differences in data collection and healthcare definitions may affect direct comparisons.
Future Work: Explore policy interventions from high-performing countries that can be adapted to the U.S. context.
Conclusion Statement: Healthcare quality in the U.S. is subpar compared to other high-income countries. Systemic reforms, not just healthcare spending, are needed to improve health outcomes.
Markdown Format:
Body Text:
Metadata:
Paper ID: pap001
Title: Comparative Analysis of U.S. Healthcare Quality
Author: Dr. Jane Smith
Publication Year: 2023
Source: Peterson-KFF Health System Tracker
Document Type: Policy Report
Topic Category: International Health System Comparison

Conclusion Statement: Healthcare quality in the U.S. is subpar compared to other high-income countries. Systemic reforms, not just healthcare spending, are needed to improve health outcomes.
Conclusion: The program significantly improved math proficiency and learner engagement. It can be replicated with adjusted instructional

```
strategies.
Implication: Improved arithmetic skills and better engagement suggest the program is effective and scalable.
Thematic Focus: International Health System Comparison | Maternal and Preventable Mortality
Limitations: Data reliability and comparability across countries may limit direct conclusions.
Future Work: Continued analysis of international healthcare system structures and their impact on population health.
Conclusion Statement: Healthcare quality in the U.S. is subpar compared to other high-income countries. Systemic reforms, not just healthcare spending, are needed to improve health outcomes.
Markdown Format:
Body Text:
Metadata:
Paper ID: pap002
Title: International Educational Equity: Lessons from High-Performing Systems
Author: Prof. John Doe
Publication Year: 2021
Source: OECD Education Working Papers, No. 229
Document Type: Policy Study
Topic Category: Educational Equity, International Policy

Conclusion: The program significantly improved math proficiency and learner engagement. It can be replicated with adjusted instructional strategies.
Conclusion: The policy highlights the importance of comparing healthcare systems and emphasizes the need for comprehensive reform.
Conclusion Statement: Healthcare quality in the U.S. is

# Save your fine-tuned model to a local directory
model_save_path = "./PH-Llama-3.1"
trainer.save_model(model_save_path)

tokenizer.save_pretrained(model_save_path)

('./PH-Llama-3.1/tokenizer_config.json',
 './PH-Llama-3.1/special_tokens_map.json',
 './PH-Llama-3.1/tokenizer.json')

torch.save(peft_model.state_dict(), "./model/PH-Llama-3.1.pth")

from huggingface_hub import HfApi, HfFolder, Repository

from huggingface_hub import login
login(token="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")

trainer.push_to_hub("iyashnayi/PH-Llama-3.1")
```

{"model_id":"1d14d80dbf784ef2a8bfa8a69f035821","version_major":2,"version_minor":0}

{"model_id":"b440cced7689413fb7a7942fda4406fb","version_major":2,"version_minor":0}

{"model_id":"70ad64ff4aaa419e97a90d4182f464cc","version_major":2,"version_minor":0}

{"model_id":"4ebe0fa5fb874390944c17e87528738d","version_major":2,"version_minor":0}

```
CommitInfo(commit_url='https://huggingface.co/iyashnayi/PH-Llama-3.1/
commit/1a0aac8f49cb258a7df6f6c2a2e37690f42bca0b',
commit_message='iyashnayi/PH-Llama-3.1', commit_description='',
oid='1a0aac8f49cb258a7df6f6c2a2e37690f42bca0b', pr_url=None,
repo_url=RepoUrl('https://huggingface.co/iyashnayi/PH-Llama-3.1',
endpoint='https://huggingface.co', repo_type='model',
repo_id='iyashnayi/PH-Llama-3.1'), pr_revision=None, pr_num=None)
```