Personalized Healthcare QA System (Chatbot) - V3 - MedDialog IC

Ed:215279167

Objective:

Develop a healthcare chatbot that answers patient inquiries on medications, symptoms, and treatments while reducing hallucinations.

Update and changes:

This updated notebook reflects changes learned from weeks 1 -7 experiments with fine-tuning. Main changes include:

- Revert to using MedDialog dataset to validate model is being fine tuned correctly

Datasets:

- Med Dialog IC - https://huggingface.co/datasets/lighteval/med_dialog

Task Breakdown:

1. Train models on medical QA datasets.
2. Fine-tune for patient-friendly responses (simplified, clear language).
3. Ensure context-aware, regulatory-compliant answers:
   o Implement FDA/TGA guideline alignment. o Develop a retrieval-based validation for generated answers.
4. Deploy as a chatbot interface (React-based UI + API integration).
5. Implement real-time fact-checking:
   o Confidence score visualization (green = high confidence, yellow = medium, red = low confidence). o Integration with PubMed and trusted medical sources.

Models to Use:

• Llama-2 7B

Evaluation Metrics:

- Rouge, BLEU, Meteor, BertScore, Manual inspection

---

1. Import libraries and model to prepare for fine tuning

```
!pip install datasets requests bitsandbytes accelerate peft trl sentencepiece wandb transformers evaluate rouge_score bert-score
```

```
Collecting datasets
  Downloading datasets-3.6.0-py3-none-any.whl.metadata (19 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (2.32.3)
Collecting bitsandbytes
  Downloading bitsandbytes-0.45.5-py3-none-manylinux_2_24_x86_64.whl.metadata (5.0 kB)
Requirement already satisfied: accelerate in /usr/local/lib/python3.11/dist-packages (1.6.0)
Requirement already satisfied: peft in /usr/local/lib/python3.11/dist-packages (0.15.2)
Collecting trl
  Downloading trl-0.17.0-py3-none-any.whl.metadata (12 kB)
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.11/dist-packages (0.2.0)
Requirement already satisfied: wandb in /usr/local/lib/python3.11/dist-packages (0.19.10)
Requirement already satisfied: transformers in /usr/local/lib/python3.11/dist-packages (4.51.3)
Collecting evaluate
  Downloading evaluate-0.4.3-py3-none-any.whl.metadata (9.2 kB)
Collecting rouge_score
  Downloading rouge_score-0.1.2.tar.gz (17 kB)
  Preparing metadata (setup.py) ... done
Collecting bert-score
  Downloading bert_score-0.3.13-py3-none-any.whl.metadata (15 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from datasets) (3.18.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages (from datasets) (2.0.2)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (18.1.0)
Collecting dill<0.3.9,>=0.3.0 (from datasets)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: tqdm>=4.66.3 in /usr/local/lib/python3.11/dist-packages (from datasets) (4.67.1)
Collecting xxhash (from datasets)
  Downloading xxhash-3.5.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (12 kB)
Collecting multiprocess<0.70.17 (from datasets)
  Downloading multiprocess-0.70.16-py311-none-any.whl.metadata (7.2 kB)
Collecting fsspec<=2025.3.0,>=2023.1.0 (from fsspec[http]<=2025.3.0,>=2023.1.0->datasets)
  Downloading fsspec-2025.3.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: huggingface-hub>=0.24.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (0.30.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from datasets) (24.2)
```

```
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-packages (from datasets) (6.0.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests) (2.4.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests) (2025.4.26)
Requirement already satisfied: torch<3,>=2.0 in /usr/local/lib/python3.11/dist-packages (from bitsandbytes) (2.6.0+cu124)
Requirement already satisfied: psutil in /usr/local/lib/python3.11/dist-packages (from accelerate) (5.9.5)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11/dist-packages (from accelerate) (0.5.3)
Requirement already satisfied: rich in /usr/local/lib/python3.11/dist-packages (from trl) (13.9.4)
Requirement already satisfied: click!=8.0.0,>=7.1 in /usr/local/lib/python3.11/dist-packages (from wandb) (8.1.8)
Requirement already satisfied: docker-pycreds>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from wandb) (0.4.0)
Requirement already satisfied: gitpython!=3.1.29,>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from wandb) (3.1.44)
Requirement already satisfied: platformdirs in /usr/local/lib/python3.11/dist-packages (from wandb) (4.3.7)
Requirement already satisfied: protobuf!=4.21.0,!=5.28.0,<7,>=3.19.0 in /usr/local/lib/python3.11/dist-packages (from wandb) (5.29
Requirement already satisfied: pydantic<3 in /usr/local/lib/python3.11/dist-packages (from wandb) (2.11.4)
Requirement already satisfied: sentry-sdk>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from wandb) (2.27.0)
Requirement already satisfied: setproctitle in /usr/local/lib/python3.11/dist-packages (from wandb) (1.3.6)
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from wandb) (75.2.0)
Requirement already satisfied: typing-extensions<5,>=4.4 in /usr/local/lib/python3.11/dist-packages (from wandb) (4.13.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.21.1)
Requirement already satisfied: absl-py in /usr/local/lib/python3.11/dist-packages (from rouge_score) (1.4.0)
Requirement already satisfied: nltk in /usr/local/lib/python3.11/dist-packages (from rouge_score) (3.9.1)
```

IMPORTANT: Restart Colab runtime after PIP install!!!

2. Implement base model and imports

First step is to import all necessary libraries and implement base model, to validate it can generate a response from prompt.

```
from google.colab import userdata
from huggingface_hub import login
from transformers import (
    AutoTokenizer,
    AutoModelForCausalLM,
    BitsAndBytesConfig,
    TrainingArguments,
    logging,
    pipeline
)
from peft import LoraConfig, PeftModel, prepare_model_for_kbit_training, get_peft_model, TaskType
from trl import SFTTrainer, SFTConfig, DataCollatorForCompletionOnlyLM
from datasets import load_dataset, Dataset, DatasetDict
from datetime import datetime
import torch
import wandb
import pandas as pd
import os
import random
import re
import matplotlib.pyplot as plt
from tqdm.notebook import tqdm
import evaluate
import numpy as np
from bert_score import BERTScorer
```

```
model_name = "meta-llama/Llama-2-7b-chat-hf"
project_name = "ed_medical"
hf_username = "digitalblue"
```

```
# log into hugging face and wandb
hf_token = userdata.get('HF_TOKEN')
login(hf_token)

wandb_api_key = userdata.get('WANDB_API_KEY')
os.environ["WANDB_API_KEY"] = wandb_api_key
wandb.login()

# Configure Weights & Biases to record against our project
os.environ["WANDB_PROJECT"] = "ed_medical"
os.environ["WANDB_LOG_MODEL"] = "checkpoint" if True else "end"
os.environ["WANDB_WATCH"] = "gradients"
```

```
wandb: Currently logged in as: digitalblue (digitalblue-ai) to https://api.wandb.ai. Use `wandb login --relogin` to force relogin
```

```
# quantisation config to use less memory when loading model
quant_config = BitsAndBytesConfig(
    load_in_4bit=True,
```

```
    bnb_4bit_use_double_quant=True,
    bnb_4bit_compute_dtype=torch.bfloat16,
    bnb_4bit_quant_type="nf4"
)
```

```
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    device_map="auto",
    quantization_config=quant_config)
```

```
config.json: 100% 614/614 [00:00<00:00, 64.1kB/s]
model.safetensors.index.json: 100% 26.8k/26.8k [00:00<00:00, 2.70MB/s]
Fetching 2 files: 100% 2/2 [01:46<00:00, 106.82s/it]
model-00002-of-00002.safetensors: 100% 3.50G/3.50G [00:49<00:00, 108MB/s]
model-00001-of-00002.safetensors: 100% 9.98G/9.98G [01:46<00:00, 280MB/s]
Loading checkpoint shards: 100% 2/2 [01:14<00:00, 33.93s/it]
generation_config.json: 100% 188/188 [00:00<00:00, 19.6kB/s]
```

```
memory = model.get_memory_footprint() / 1e6
print(f"Memory footprint: {memory:,.1f} MB")
```

```
Memory footprint: 3,762.8 MB
```

```
# model architecture
model
```

```
LlamaForCausalLM(
  (model): LlamaModel(
    (embed_tokens): Embedding(32000, 4096)
    (layers): ModuleList(
      (0-31): 32 x LlamaDecoderLayer(
        (self_attn): LlamaAttention(
          (q_proj): Linear4bit(in_features=4096, out_features=4096, bias=False)
          (k_proj): Linear4bit(in_features=4096, out_features=4096, bias=False)
          (v_proj): Linear4bit(in_features=4096, out_features=4096, bias=False)
          (o_proj): Linear4bit(in_features=4096, out_features=4096, bias=False)
        )
        (mlp): LlamaMLP(
          (gate_proj): Linear4bit(in_features=4096, out_features=11008, bias=False)
          (up_proj): Linear4bit(in_features=4096, out_features=11008, bias=False)
          (down_proj): Linear4bit(in_features=11008, out_features=4096, bias=False)
          (act_fn): SiLU()
        )
        (input_layernorm): LlamaRMSNorm((4096,), eps=1e-05)
        (post_attention_layernorm): LlamaRMSNorm((4096,), eps=1e-05)
      )
    )
    (norm): LlamaRMSNorm((4096,), eps=1e-05)
    (rotary_emb): LlamaRotaryEmbedding()
  )
  (lm_head): Linear(in_features=4096, out_features=32000, bias=False)
)
```

```
# initialise tokenizer and pipeline
tokenizer = AutoTokenizer.from_pretrained(model_name, token=hf_token)
tokenizer.pad_token = tokenizer.eos_token
tokenizer.padding_side = "right"
```

```
tokenizer_config.json: 100% 1.62k/1.62k [00:00<00:00, 166kB/s]
tokenizer.model: 100% 500k/500k [00:00<00:00, 11.7MB/s]
tokenizer.json: 100% 1.84M/1.84M [00:00<00:00, 7.43MB/s]
special_tokens_map.json: 100% 414/414 [00:00<00:00, 35.0kB/s]
```

```
messages = [
    {"role": "system", "content": "You are a helpful personalised medical assistant"},
    {"role": "user", "content": "How can I get rid of the flu?"}
  ]
```

```
# verify base model generates response
inputs = tokenizer.apply_chat_template(messages, return_tensors="pt").to("cuda")
outputs = model.generate(inputs, max_new_tokens=250)
print(tokenizer.decode(outputs[0]))
```

```
<s> [INST] <<SYS>>
You are a helpful personalised medical assistant
<</SYS>>

How can I get rid of the flu? [/INST]  Hello there! I'm glad you asked! 😊 As a helpful personalized medical assistant, I'm here to

First and foremost, it's important to understand that the flu is a viral infection, and there's no magic pill or potion that can mak

1. Stay hydrated: Drink plenty of fluids, such as water, tea, or soup, to help loosen congestion and replace lost fluids. Aim for at
2. Rest: Give your body the rest it needs to fight off the infection. Try to get at least 8 hours of sleep each night and take naps
3. Use over-the-counter medications: Over-the-counter medications like acetaminophen (Tylenol) or ibuprofen (Advil, Motrin) can help
```

3. Dataset analysis and preprocessing for Llama

The datasets need to be analysed and prepared for fine-tuning the model. This includes removing invalid data and formatting into the required format with special tokens the Llama 2 model requires. Dataset acquired from HuggingFace

- MedDialog_IC - https://huggingface.co/datasets/lighteval/med_dialog

```
# format input from datasets into prompt format required by llama model
def format_llama_prompt(user_message, model_answer):
  prompt = '<s>[INST] ' # special token - commence instruct
  prompt += user_message.strip()
  prompt += ' [/INST] ' # special token - end instruct
  prompt += model_answer.strip()
  prompt += ' </s>'# special token - end
  return prompt

def format_llama_prompt_with_context(user_message, model_answer, context):
  prompt = '<s>[INST] Given this context: ' # special token - commence instruct
  prompt += context['contexts'][0].strip()
  prompt += ' Question: '
  prompt += user_message.strip()
  prompt += ' [/INST] ' # special token - end instruct
  prompt += model_answer.strip()
  prompt += ' </s>'# special token - end
  return prompt
```

```
# load med_dialog dataset
med_dialog_ic = load_dataset("lighteval/med_dialog", "icliniq")
```

```
0000.parquet: 100%  12.2M/12.2M [00:00<00:00, 20.5MB/s]
0000.parquet: 100%  1.51M/1.51M [00:00<00:00, 19.8MB/s]
0000.parquet: 100%  1.50M/1.50M [00:00<00:00, 16.1MB/s]
Generating train split: 100%  24851/24851 [00:00<00:00, 61193.22 examples/s]
Generating validation split: 100%  3105/3105 [00:00<00:00, 31569.68 examples/s]
Generating test split: 100%  3108/3108 [00:00<00:00, 42613.92 examples/s]
```

```
med_dialog_ic
```

```
DatasetDict({
    train: Dataset({
        features: ['tgt', 'src', 'id'],
        num_rows: 24851
    })
    validation: Dataset({
        features: ['tgt', 'src', 'id'],
        num_rows: 3105
    })
    test: Dataset({
        features: ['tgt', 'src', 'id'],
        num_rows: 3108
    })
})
```

```
med_dialog_ic['train'][100]
```

```
{'tgt': 'What is the small yellow bump above my tonsil?',
 'src': 'Patient: Hi doctor, I have a small round yellow bump or circle on the back of the arch, right above my tonsil. It is there for a month. It does not show any changes. I do not have any symptoms or soreness. Doctor: Hi. I have gone through the attachment (attachment removed to protect patient identity). It appears like a small cyst. There can be a fluid collection inside the minor salivary gland. You need not worry about it as it goes down on its own. However, if you have any symptoms, you may need to address it. Since you do not have any signs, we need not worry about it. Occasionally, it could be a tonsillolith which is stone formation inside the naturally occurring crypts of the tonsil. You need not worry unless it is symptomatic. Follow up after six months regarding this yellow spot. For more information consult an ENT otolaryngologist online --> https://www.icliniq.com/ask-a-doctor-
```

```
online/ENT-Otolaryngologist',
 'id': 21325}
```

```
# med_dialog data requires prep to parse patient/doctor chat in src
def prep_med_dialog(src_dialog):
  init_split = src_dialog.split('Patient: ')
  init_split = [item for item in init_split if item] # remove empty item introduced by init split
  if len(init_split) > 2:
    print('Longer dialog: ' + str(init_split))
    return None
  else:
    #print(next_split)
    next_split = init_split[0].split(' Doctor:')
    if len(next_split) > 1:
      return format_llama_prompt(next_split[0], next_split[1])
    else:
      print("Invalid dialog: " + str(src_dialog))
      return None

```

```
# create subset of med_dialag src value
med_dialog_ic_sub = med_dialog_ic['train']['src']
```

```
# create list of formatted prompts
med_dialog_ic_as_prompt = []
for index, dialog in enumerate(med_dialog_ic_sub):
  med_dialog_ic_as_prompt.append(prep_med_dialog(dialog))
```

```
med_dialog_ic_as_prompt[100]
```

```
'<s>[INST] Hi doctor, I have a small round yellow bump or circle on the back of the arch, right above my tonsil. It is there for a month. It does not show any changes. I do not have any symptoms or soreness. [/INST] Hi. I have gone through the attachment (attachment removed to protect patient identity). It appears like a small cyst. There can be a fluid collection inside the minor salivary gland. You need not worry about it as it goes down on its own. However, if you have any symptoms, you may need to address it. Since you do not have any signs, we need not worry about it. Occasionally, it could be a tonsillolith which is stone formation inside the
```

```
print(len(med_dialog_ic_as_prompt))
print(med_dialog_ic_as_prompt[0])
```

```
24851
<s>[INST] Hello doctor, My friend aged 30 had two drops of phenol mistaking for milk. He vomited and had lot of salt water. Please a
```

```
# collect all formatted data into one dataset
prompt_dataset = med_dialog_ic_as_prompt
print(len(prompt_dataset))
```

```
24851
```

```
# convert to huggingface dataset, create splits and push to hub
prompt_hf_dataset = Dataset.from_dict({"text": prompt_dataset})
```

```
ds_train = prompt_hf_dataset.train_test_split(test_size=0.2, seed=42)
```

```
ds_train
```

```
DatasetDict({
    train: Dataset({
        features: ['text'],
        num_rows: 19880
    })
    test: Dataset({
        features: ['text'],
        num_rows: 4971
    })
})
```

```
ds_test = ds_train['test'].train_test_split(test_size=0.5, seed=42)
ds_test
```

```
DatasetDict({
    train: Dataset({
        features: ['text'],
        num_rows: 2485
    })
    test: Dataset({
        features: ['text'],
        num_rows: 2486
```

```
        })
    })
```

```
ds_splits = DatasetDict({
    'train': ds_train['train'],
    'validation': ds_test['train'],
    'test': ds_test['test']
})
```

```
ds_splits
```

```
DatasetDict({
    train: Dataset({
        features: ['text'],
        num_rows: 19880
    })
    validation: Dataset({
        features: ['text'],
        num_rows: 2485
    })
    test: Dataset({
        features: ['text'],
        num_rows: 2486
    })
})
```

```
# push to hugging face
ds_splits.push_to_hub(f"{hf_username}/{project_name}-med-dialog-ic", private=True)
```

```
Uploading the dataset shards: 100%  1/1 [00:01<00:00,  1.74s/it]
Creating parquet from Arrow format: 100%  20/20 [00:00<00:00, 24.17ba/s]
Uploading the dataset shards: 100%  1/1 [00:00<00:00,  1.49it/s]
Creating parquet from Arrow format: 100%  3/3 [00:00<00:00, 17.92ba/s]
Uploading the dataset shards: 100%  1/1 [00:00<00:00,  1.44it/s]
Creating parquet from Arrow format: 100%  3/3 [00:00<00:00, 16.33ba/s]
CommitInfo(commit_url='https://huggingface.co/datasets/digitalblue/ed_medical-med-dialog-ic/commit/f97aa80aa05fc6888b489d55ca90b4f68c8ee458', commit_message='Upload dataset', commit_description='', oid='f97aa80aa05fc6888b489d55ca90b4f68c8ee458', pr_url=None,
```

4. Model Training Pipeline

The model training pipeline is setup for fine-tuning. During this process additional data processing was required to filter out data with execessive token length that would exhaust GPU resources.

Initially four variants(A,B,C,D) of the model were fine-tuned on combination dataset, however focus is now on a single model trained on Med Dialog IC dataset. It maintains integration with HuggingFace and Weight and Bias platform so models and runs are saved for later retrieval.

The models were fine-tuned on the **train** set, and evaluated with **validation** set.

```
qa_dataset = load_dataset("digitalblue/ed_medical-med-dialog-ic")
qa_dataset
```

```
README.md: 100%  527/527 [00:00<00:00, 54.1kB/s]
train-00000-of-00001.parquet: 100%  9.05M/9.05M [00:00<00:00, 28.9MB/s]
validation-00000-of-00001.parquet: 100%  1.14M/1.14M [00:00<00:00, 58.1MB/s]
test-00000-of-00001.parquet: 100%  1.17M/1.17M [00:00<00:00, 21.6MB/s]
Generating train split: 100%  19880/19880 [00:00<00:00, 133999.04 examples/s]
Generating validation split: 100%  2485/2485 [00:00<00:00, 65229.62 examples/s]
Generating test split: 100%  2486/2486 [00:00<00:00, 80844.18 examples/s]
DatasetDict({
    train: Dataset({
        features: ['text'],
        num_rows: 19880
    })
    validation: Dataset({
        features: ['text'],
        num_rows: 2485
    })
    test: Dataset({
        features: ['text'],
        num_rows: 2486
    })
})
```

```
# getting token lengths from training dataset
train_length = []
train_token_length = []
for item in qa_dataset['train']:
  if item['text']:
    train_length.append(len(item['text']))
    tokens = tokenizer.encode(item['text'])
    train_token_length.append(len(tokens))
  else:
    print(item)
```

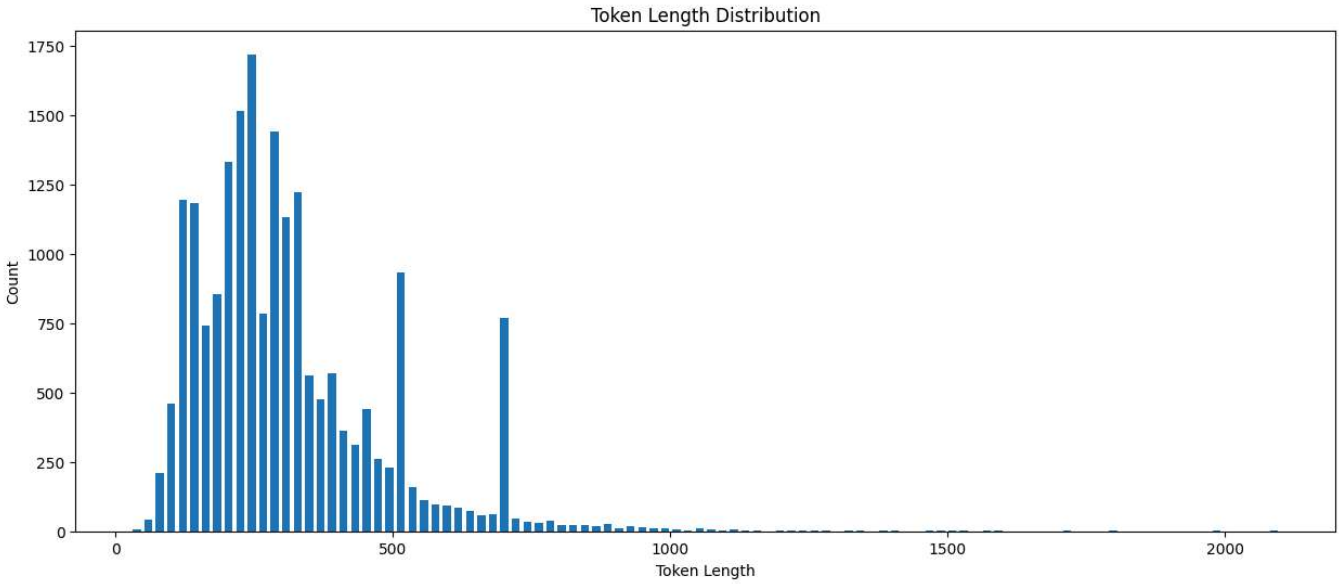
```
len(train_token_length)
```

```
19880
```

```
# print the min, max and average character lengths and token count lengths for training data
print(min(train_length), max(train_length), sum(train_length)/len(train_length))
print(min(train_token_length), max(train_token_length), sum(train_token_length)/len(train_token_length))
```

```
84 8371 1173.2094064386317
28 2098 311.1887323943662
```

```
# plot the token lengths
plt.figure(figsize=(15, 6))
plt.hist(train_token_length, rwidth=0.7, bins=100)
plt.xlabel('Token Length')
plt.ylabel('Count')
plt.title('Token Length Distribution')
plt.show()
```




```
# avg is now approx 311, but will keep 350 tokens, therefore will create new datasets
# to filter out any rows above this token count
MAX_TOKENS = 350
DATASET_SIZE = 1000
def get_filtered_dataset(dataset, max_tokens, size):
  count = 0
  train_size = round(size * 0.9) # 90% for train
  valid_size = round(size * 0.1) # 10% for eval
  print(f"Train size: {train_size}, Eval size: {valid_size}")

  filtered_train_dataset = []
  for item in dataset['train']:
    if item['text']:
      tokens = tokenizer.encode(item['text'])
      if len(tokens) < max_tokens:
        filtered_train_dataset.append(item)
        count += 1
      if count >= train_size:
        break

  count = 0
  filtered_eval_dataset = []
  for item in dataset['validation']:
    if item['text']:
      tokens = tokenizer.encode(item['text'])
      if len(tokens) < max_tokens:
        filtered_eval_dataset.append(item)

        count += 1
      if count >= valid_size:
        break

  return filtered_train_dataset, filtered_eval_dataset
```

```
qa_train, qa_val = get_filtered_dataset(qa_dataset, MAX_TOKENS, DATASET_SIZE)
qa_train = Dataset.from_list(qa_train)
qa_val = Dataset.from_list(qa_val)
print(qa_train)
print(qa_val)
```

```
Train size: 900, Eval size: 100
Dataset({
    features: ['text'],
    num_rows: 900
})
Dataset({
    features: ['text'],
    num_rows: 100
})
```

```
qa_val[10]
```

```
{'text': '<s>[INST] Hello doctor, My friend aged 30 had two drops of phenol mistaking for milk. He vomited and had lot of salt water. Please advice for any side effect. [/INST] Hi. I want to assure you not to worry as everything is going to be fine if proper care and treatment is opted in for. I have thoroughly gone through your case and can well understand your genuine health concerns. 1. No, there is not much problem right now as he vomited and also had a lot of water. It is fine because it was only two drops. 2. We usually do not go for emesis (vomiting) for phenol poisoning cases. Because, it is a volatile compound and causes vapors entering the lungs through the airways. 3. As it was only two drops, I do not think it may cause much trouble. If he feels short of breath contact me back, otherwise fine. 1. He should avoid re-exposure. 2. Rule out if there is some sort of suicidal ideation due to depression or anxiety and take him to a psychiatrist for physical evaluation. For further follow up consult a general practitioner online.---> https://www.icliniq.com/ask-a-doctor-online/general-practitioner </s>'}
```

```
# set constants
MAX_SEQUENCE_LENGTH = 350 # calculated from avg token lenght in dataset

# Run name for saving the model in the hub
RUN_NAME =  f"{datetime.now():%Y-%m-%d_%H.%M.%S}"
PROJECT_RUN_NAME = f"{project_name}-{RUN_NAME}"
HUB_MODEL_NAME = f"{hf_username}/{PROJECT_RUN_NAME}"

# qlora hyper params
LORA_R = 16 # Reduce LoRA rank (lower = less memory) , initial = 32
LORA_ALPHA = 32 # Lower alpha , initial = 64
TARGET_MODULES = ["q_proj", "k_proj", "v_proj", "o_proj"]
LORA_DROPOUT = 0.05
QUANT_4_BIT = True

# training hyper params
EPOCHS = 3
BATCH_SIZE = 4  # Reduce batch size , initial = 4
GRADIENT_ACCUMULATION_STEPS = 8  # Simulate batch size 8, initial = 1
LEARNING_RATE = 2e-4
LR_SCHEDULER_TYPE = 'cosine'
WARMUP_RATIO = 0.03
OPTIMIZER = "paged_adamw_32bit"
STEPS = 20
SAVE_STEPS = 2000
LOG_TO_WANDB = True
```

```
response_template = " [/INST] "
collator = DataCollatorForCompletionOnlyLM(response_template, tokenizer=tokenizer)
```

```
lora_params = LoraConfig(
    r=LORA_R,
    lora_alpha=LORA_ALPHA,
    lora_dropout=LORA_DROPOUT,
    target_modules=TARGET_MODULES,
    bias="none",
    task_type="CAUSAL_LM"
)
```

```
training_params = SFTConfig(
    output_dir=PROJECT_RUN_NAME,
    num_train_epochs=EPOCHS,
    per_device_train_batch_size=BATCH_SIZE,
    per_device_eval_batch_size=1,
    #eval_strategy="no",
    gradient_accumulation_steps=GRADIENT_ACCUMULATION_STEPS,
    optim=OPTIMIZER,
    save_steps=SAVE_STEPS,
    save_total_limit=10,
    logging_steps=STEPS,
    learning_rate=LEARNING_RATE,
    weight_decay=0.001,
    fp16=False,
    bf16=True,
    max_grad_norm=0.3,
    max_steps=-1,
    warmup_ratio=WARMUP_RATIO,
    group_by_length=True,
    lr_scheduler_type=LR_SCHEDULER_TYPE,
    report_to="wandb" if True else None,
    run_name=RUN_NAME,
    max_seq_length=MAX_SEQUENCE_LENGTH,
    dataset_text_field="text",
    save_strategy="steps",
    hub_strategy="every_save",
    push_to_hub=True,
    hub_model_id=HUB_MODEL_NAME,
    hub_private_repo=True
```

```
    #neftune_noise_alpha=5 # using NEFTune as describe in SFT Trainer docs for increased conversational quality
)
```

```
!nvidia-smi
```

```
Mon May 12 02:32:01 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.54.15              Driver Version: 550.54.15      CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  Tesla T4                       Off |   00000000:00:04.0 Off |                    0 |
| N/A   37C    P0             25W /   70W |     104MiB /  15360MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                              GPU Memory |
|        ID   ID                                                               Usage      |
|=========================================================================================|
+-----------------------------------------------------------------------------------------+
```

```
# clear gpu memory
torch.cuda.empty_cache()
torch.cuda.reset_max_memory_allocated()
```

```
/usr/local/lib/python3.11/dist-packages/torch/cuda/memory.py:391: FutureWarning: torch.cuda.reset_max_memory_allocated now calls tor
  warnings.warn(
```

```
# Prepare the model for k-bit training (important for 4-bit)
model = prepare_model_for_kbit_training(model)
```

```
model = get_peft_model(model, lora_params)
```

```
trainer = SFTTrainer(
    model=model,
    train_dataset=qa_train,
    eval_dataset=qa_val,
    peft_config=lora_params,
    args=training_params
)

```

```
Converting train dataset to ChatML: 100% 900/900 [00:00<00:00, 20163.63 examples/s]
Adding EOS to train dataset: 100% 900/900 [00:00<00:00, 16423.06 examples/s]
Tokenizing train dataset: 100% 900/900 [00:00<00:00, 1561.17 examples/s]
Truncating train dataset: 100% 900/900 [00:00<00:00, 42134.05 examples/s]
Converting eval dataset to ChatML: 100% 100/100 [00:00<00:00, 3621.24 examples/s]
Adding EOS to eval dataset: 100% 100/100 [00:00<00:00, 5019.15 examples/s]
Tokenizing eval dataset: 100% 100/100 [00:00<00:00, 1041.38 examples/s]
Truncating eval dataset: 100% 100/100 [00:00<00:00, 7792.34 examples/s]
No label_names provided for model class `PeftModelForCausalLM`. Since `PeftModel` hides base models input arguments, if label_names
```

Double-click (or enter) to edit

```
#torch.cuda.empty_cache()
trainer.train()
trainer.model.push_to_hub(PROJECT_RUN_NAME, private=True)
print(f"Model saved to HuggingFace as: {PROJECT_RUN_NAME}")
model.save_pretrained(PROJECT_RUN_NAME)

```

Changes to your `wandb` environment variables will be ignored because your `wandb` session has already started. For more information on how to modify your settings with `wandb.init()` arguments, please refer to the W&B docs.
Tracking run with wandb version 0.19.10
Run data is saved locally in `/content/wandb/run-20250512_023609-ralcfsyz`
Syncing run **2025-05-12_02.31.47** to Weights & Biases (docs)
View project at https://wandb.ai/digitalblue-ai/ed_medical
View run at https://wandb.ai/digitalblue-ai/ed_medical/runs/ralcfsyz

```
wandb: WARNING The get_url method is deprecated and will be removed in a future release. Please use `run.url` instead.
`use_cache=True` is incompatible with gradient checkpointing. Setting `use_cache=False`.
/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745: UserWarning: torch.utils.checkpoint: the use_reentrant para
  return fn(*args, **kwargs)
```

[84/84 3:21:55, Epoch 2/3]

Step	Training Loss
20	2.291700
40	1.684000
60	1.459900
80	1.404900

```
wandb: Adding directory to artifact (./ed_medical-2025-05-12_02.31.47/checkpoint-84)... Done. 0.8s
No label_names provided for model class `PeftModelForCausalLM`. Since `PeftModel` hides base models input arguments, if label_names
```

README.md: 100% 1.52k/1.52k [00:00<00:00, 99.9kB/s]

```
No files have been modified since last commit. Skipping to prevent empty commit.
WARNING:huggingface_hub.hf_api:No files have been modified since last commit. Skipping to prevent empty commit.
Model saved to HuggingFace as: ed_medical-2025-05-12_02.31.47
```

```
1 results = trainer.evaluate()
2 print(results)
3
```

[100/100 03:01]

```
{'eval_loss': 1.3957768678665161, 'eval_runtime': 184.2312, 'eval_samples_per_second': 0.543, 'eval_steps_per_second': 0.543}
```

```
1 wandb.finish()
```

Run history:

eval/loss
eval/runtime
eval/samples_per_second
eval/steps_per_second
train/epoch
train/global_step
train/grad_norm
train/learning_rate
train/loss
train/mean_token_accuracy
train/num_tokens

Run summary:

eval/loss	1.39578
eval/runtime	184.2312
eval/samples_per_second	0.543
eval/steps_per_second	0.543
total_flos	2.4278144750813184e+16
train/epoch	2.92444
train/global_step	84
train/grad_norm	0.52332
train/learning_rate	0.0
train/loss	1.4049
train/mean_token_accuracy	0.66853
train/num_tokens	626881
train_loss	1.69449
train_runtime	12307.6128
train_samples_per_second	0.219
train_steps_per_second	0.007

View run **2025-05-12_02.31.47** at: https://wandb.ai/digitalblue-ai/ed_medical/runs/ralcfsyz
View project at: https://wandb.ai/digitalblue-ai/ed_medical
Synced 5 W&B file(s), 0 media file(s), 17 artifact file(s) and 0 other file(s)
Find logs at: `./wandb/run-20250512_023609-ralcfsyz/logs`

```
!zip -r ed_medical-2025-05-12_02.31.47.zip ed_medical-2025-05-12_02.31.47
```

```
  adding: ed_medical-2025-05-12_02.31.47/ (stored 0%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/ (stored 0%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/rng_state.pth (deflated 25%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/adapter_model.safetensors (deflated 7%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/training_args.bin (deflated 51%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/optimizer.pt (deflated 8%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/trainer_state.json (deflated 63%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/special_tokens_map.json (deflated 74%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/tokenizer.model (deflated 55%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/scheduler.pt (deflated 57%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/tokenizer.json (deflated 85%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/tokenizer_config.json (deflated 66%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/adapter_config.json (deflated 55%)
  adding: ed_medical-2025-05-12_02.31.47/checkpoint-84/README.md (deflated 66%)
  adding: ed_medical-2025-05-12_02.31.47/adapter_model.safetensors (deflated 7%)
  adding: ed_medical-2025-05-12_02.31.47/training_args.bin (deflated 51%)
  adding: ed_medical-2025-05-12_02.31.47/special_tokens_map.json (deflated 74%)
  adding: ed_medical-2025-05-12_02.31.47/tokenizer.model (deflated 55%)
  adding: ed_medical-2025-05-12_02.31.47/tokenizer.json (deflated 85%)
  adding: ed_medical-2025-05-12_02.31.47/tokenizer_config.json (deflated 66%)
  adding: ed_medical-2025-05-12_02.31.47/adapter_config.json (deflated 55%)
  adding: ed_medical-2025-05-12_02.31.47/README.md (deflated 45%)
```

```
!ls -la
```

```
total 243168
drwxr-xr-x 1 root root      4096 May 12 06:04 .
drwxr-xr-x 1 root root      4096 May 12 02:24 ..
drwxr-xr-x 4 root root      4096 May  8 13:38 .config
drwxr-xr-x 3 root root      4096 May 12 06:01 ed_medical-2025-05-12_02.31.47
-rw-r--r-- 1 root root 248971266 May 12 06:04 ed_medical-2025-05-12_02.31.47.zip
drwxr-xr-x 1 root root      4096 May  8 13:38 sample_data
drwxr-xr-x 3 root root      4096 May 12 02:36 wandb
```

```
!cp ed_medical-2025-05-12_02.31.47.zip /content/drive/MyDrive/
```

```
cp: cannot create regular file '/content/drive/MyDrive/': No such file or directory
```

```
# stop notebook and disconnect GPU after finishing above steps as this process can take several hours
from google.colab import runtime
runtime.unassign()
```

5. Evaluate the models - generate evaluation data

At this step the fine-tuned model created in the previous step can be re-loaded without need to re-run step 4. Generated responses will be collected, in addition to the base model to use as a benchmark.

The prompts to generate the responses are 50 samples from the **test** set which was not used for fine-tuning.

The generated response for each model are then saved to HuggingFace hub for later analysis.

```
# define models saved to huggingface hub from previous step
MODEL_A = "digitalblue/ed_medical-2025-04-03_09.11.29" # trained on 200 rows
MODEL_B = "digitalblue/ed_medical-2025-04-05_10.59.54" # trained on 800 rows
MODEL_C = "digitalblue/ed_medical-2025-04-05_11.36.32" # trained on 1600 rows
MODEL_D = "digitalblue/ed_medical-2025-04-06_10.40.29" # trained on 1600 rows with NEFTune
MODEL_E = "digitalblue/ed_medical-2025-04-15_12.43.48" # trained on 8000 rows of pubmedqa only w/ NEFTune
MODEL_F = "digitalblue/ed_medical-2025-05-05_23.51.25" # trained on 9000 row, lower learing rate, 3 epochs
MODEL_G = "digitalblue/ed_medical-2025-05-08_01.23.37" # trained on 9000 rows, NEFTune false
MODEL_H = "digitalblue/ed_medical-2025-05-11_04.05.52" # trained on 900 row, with context, 3 epochs
MODEL_I = "digitalblue/ed_medical-2025-05-12_02.31.47" # trained on 900 rows of MedDialog
```

```
# NOTE: Only using model I for MedDialog
```

```
# quantisation config to use less memory when loading model
quant_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_compute_dtype=torch.bfloat16,
    bnb_4bit_quant_type="nf4"
)
```

```
model_base = AutoModelForCausalLM.from_pretrained(
    model_name,
    quantization_config=quant_config,
    device_map="auto")
```

```
config.json: 100%  614/614 [00:00<00:00, 54.4kB/s]
model.safetensors.index.json: 100%  26.8k/26.8k [00:00<00:00, 3.37MB/s]
Fetching 2 files: 100%  2/2 [00:57<00:00, 57.65s/it]
model-00002-of-00002.safetensors: 100%  3.50G/3.50G [00:29<00:00, 122MB/s]
model-00001-of-00002.safetensors: 100%  9.98G/9.98G [00:57<00:00, 219MB/s]
Loading checkpoint shards: 100%  2/2 [00:16<00:00, 7.56s/it]
generation_config.json: 100%  188/188 [00:00<00:00, 21.4kB/s]
```

```
# initialise tokenizer and pipeline
tokenizer = AutoTokenizer.from_pretrained(model_name, token=hf_token)
tokenizer.pad_token = tokenizer.eos_token
tokenizer.padding_side = "right"
```

```
tokenizer_config.json: 100%  1.62k/1.62k [00:00<00:00, 206kB/s]
tokenizer.model: 100%  500k/500k [00:00<00:00, 20.2MB/s]
tokenizer.json: 100%  1.84M/1.84M [00:00<00:00, 4.25MB/s]
special_tokens_map.json: 100%  414/414 [00:00<00:00, 54.4kB/s]
```

```
qa_test = load_dataset("digitalblue/ed_medical-med-dialog-ic", split="test[:50]") # first 50 from test set
#qa_test = load_dataset("digitalblue/ed_medical-pubmedqa-artifical", split="test[:50]") # load only pubmedqa YES rows, test set
```

```
README.md: 100%  527/527 [00:00<00:00, 66.3kB/s]
train-00000-of-00001.parquet: 100%  9.05M/9.05M [00:00<00:00, 24.7MB/s]
validation-00000-of-00001.parquet: 100%  1.14M/1.14M [00:00<00:00, 99.5MB/s]
test-00000-of-00001.parquet: 100%  1.17M/1.17M [00:00<00:00, 1.37MB/s]
Generating train split: 100%  19880/19880 [00:00<00:00, 194488.27 examples/s]
Generating validation split: 100%  2485/2485 [00:00<00:00, 132083.56 examples/s]
Generating test split: 100%  2486/2486 [00:00<00:00, 109318.74 examples/s]
```

```
# split into question and answer list
qa_list = []
for item in qa_test:
  prompt = item['text']
  question = re.search(r'\[INST\] (.*) \[/INST\]', prompt).group(1)
  response = re.search(r'\[/INST\] (.*) \</s\>', prompt).group(1)
  qa_list.append([question, response])
```

```
qa_list[40][0] # question
```

```
'Hello doctor, My friend aged 30 had two drops of phenol mistaking for milk. He vomited and had lot of salt water. Please advice fo
```

```
prompt1 = "You are a helpful personalised medical assistant. In your response do not include any personal names and keep your answer
#prompt2 = "You are a helpful personalised medical assistant. Provide a friendly personal response to the patients medical question

def get_model_responses(model_x, qa_list):
  model_responses = []
  for item in qa_list:
    question = item[0]
    # print(item)
    # print(question)
    # print("-----")
    messages = [
      {"role": "system", "content": prompt1},
      {"role": "user", "content": question}
    ]
    inputs = tokenizer.apply_chat_template(messages, return_tensors="pt").to("cuda")
    outputs = model_x.generate(inputs, max_new_tokens=250, temperature=0.1)
    response = tokenizer.decode(outputs[0], skip_special_tokens=False)
    answer = response.split('[/INST] ')[1]  # get text after /INST token
    answer = answer.replace('</s>', '') # remove trailing special token if present
    model_responses.append(answer)
  return model_responses
```

```
model_base_responses_med_dialog = get_model_responses(model_base, qa_list)
prompt_base_med_dialog_dataset = Dataset.from_dict({"text": model_base_responses_med_dialog})
```

```
prompt_base_med_dialog_dataset.push_to_hub(f"{hf_username}/model_base_responses_med_dialog", private=True)
```

```
Uploading the dataset shards: 100% 1/1 [00:02<00:00, 2.25s/it]
Creating parquet from Arrow format: 100% 1/1 [00:00<00:00, 119.31ba/s]
CommitInfo(commit_url='https://huggingface.co/datasets/digitalblue/model_base_responses_med_dialog/commit/287eeed90021aec9c859caa63f
commit_message='Upload dataset', commit_description='', oid='287eeed90021aec9c859caa63f68ea707322ce26', pr_url=None,
repo_url=RepoUrl('https://huggingface.co/datasets/digitalblue/model_base_responses_med_dialog', endpoint='https://huggingface.co',
```

```
model_i = PeftModel.from_pretrained(model_base, MODEL_I)
```

```
adapter_config.json: 100% 813/813 [00:00<00:00, 103kB/s]
adapter_model.safetensors: 100% 67.1M/67.1M [00:02<00:00, 22.6MB/s]
```

```
print(model_base.lm_head.weight[1000, :20])
print(model_i.lm_head.weight[1000, :20])
```

```
tensor([ 0.0009, -0.0043,  0.0259, -0.0030, -0.0270,  0.0030,  0.0097,  0.0096,
        -0.0035, -0.0179,  0.0415,  0.0315, -0.0107,  0.0056,  0.0121,  0.0184,
         0.0088, -0.0099,  0.0195, -0.0232], device='cuda:0',
       dtype=torch.float16)
tensor([ 0.0009, -0.0043,  0.0259, -0.0030, -0.0270,  0.0030,  0.0097,  0.0096,
        -0.0035, -0.0179,  0.0415,  0.0315, -0.0107,  0.0056,  0.0121,  0.0184,
         0.0088, -0.0099,  0.0195, -0.0232], device='cuda:0',
       dtype=torch.float16)
```

```
model_i_responses = get_model_responses(model_i, qa_list)
prompt_i_dataset = Dataset.from_dict({"text": model_i_responses})
prompt_i_dataset.push_to_hub(f"{hf_username}/model_i_responses", private=True)
```

```
Uploading the dataset shards: 100% 1/1 [00:01<00:00, 1.32s/it]
Creating parquet from Arrow format: 100% 1/1 [00:00<00:00, 132.15ba/s]
CommitInfo(commit_url='https://huggingface.co/datasets/digitalblue/model_i_responses/commit/63b2c60c25952eea26fd92d85b45dd592c5be46
commit_message='Upload dataset', commit_description='', oid='63b2c60c25952eea26fd92d85b45dd592c5be46e', pr_url=None,
repo_url=RepoUrl('https://huggingface.co/datasets/digitalblue/model_i_responses', endpoint='https://huggingface.co',
```

```
# stop notebook and disconnect GPU after finishing above steps as this process can take several hours
from google.colab import runtime
runtime.unassign()
```

6. Evaluate fine tuned models - get metrics from generated responses

In this final step, the generated response can be reloaded from HuggingFace to evaluate and calculate metrics against the expected responses in **test** dataset.

```
qa_test = load_dataset("digitalblue/ed_medical-med-dialog-ic", split="test[:50]") # load only pubmedqa YES rows, test set
```

```
qa_list = []
for item in qa_test:
  prompt = item['text']
  question = re.search(r'\[INST\] (.*) \[/INST\]', prompt).group(1)
  response = re.search(r'\[/INST\] (.*) \</s\>', prompt).group(1)
  qa_list.append([question, response])
```

```
qa_list[40][1] # index 1 is dataset real response
```

```
'Hi. I want to assure you not to worry as everything is going to be fine if proper care and treatment is opted in for. I have thoro
ughly gone through your case and can well understand your genuine health concerns. 1. No, there is not much problem right now as he
vomited and also had a lot of water. It is fine because it was only two drops. 2. We usually do not go for emesis (vomiting) for ph
enol poisoning cases. Because, it is a volatile compound and causes vapors entering the lungs through the airways. 3. As it was onl
y two drops, I do not think it may cause much trouble. If he feels short of breath contact me back. otherwise fine. 1. He should av
```

```
qa_references = [item[1] for item in qa_list] # get list of reference responses
```

```
# load saved response data to evaluate
model_base_med_dialog_dataset = load_dataset("digitalblue/model_base_responses_med_dialog")
model_i_dataset = load_dataset("digitalblue/model_i_responses")
```

```
README.md: 100%  266/266 [00:00<00:00, 33.0kB/s]
train-00000-of-00001.parquet: 100%  27.3k/27.3k [00:00<00:00, 3.29MB/s]
Generating train split: 100%  50/50 [00:00<00:00, 4073.41 examples/s]
README.md: 100%  265/265 [00:00<00:00, 34.9kB/s]
train-00000-of-00001.parquet: 100%  4.74k/4.74k [00:00<00:00, 618kB/s]
Generating train split: 100%  50/50 [00:00<00:00, 4062.83 examples/s]
```

```
# convert to list
base_med_dialog_response_list = model_base_med_dialog_dataset['train']['text'][:50]
i_response_list = model_i_dataset['train']['text']
```

```
def calc_rouge_metric(references, predictions):
  rouge = evaluate.load('rouge')
  results = rouge.compute(
    predictions=predictions,
    references=references
  )
  return results

def calc_bleu_metric(references, predictions):
  bleu = evaluate.load('bleu')
  results = bleu.compute(
    predictions=predictions,
    references=references
  )
  return results

def calc_meteor_metric(references, predictions):
  meteor = evaluate.load('meteor')
  results = meteor.compute(
    predictions=predictions,
    references=[[ref] for ref in references]
  )
  return results

def calc_bert_score(references, predictions):
  bert_scorer = BERTScorer(model_type='bert-base-uncased')
  P = []
  R = []
  F1 = []

  for i in range(len(predictions)):
    p_i, r_i, f1_i = bert_scorer.score([predictions[i]], [references[i]])
    P.append(p_i)
    R.append(r_i)
    F1.append(f1_i)
  results = {
    'P': np.mean(P),
    'R': np.mean(R),
    'F1': np.mean(F1)
  }
  return results
```

```
rouge_base_med_dialog = calc_rouge_metric(qa_references, base_med_dialog_response_list)
rouge_i = calc_rouge_metric(qa_references, i_response_list)
```

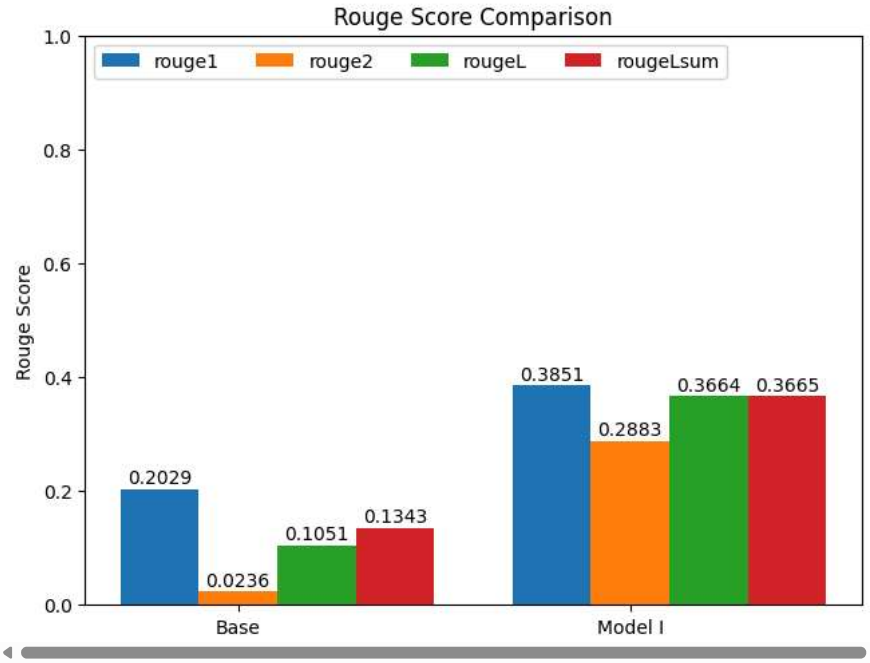
```
Downloading builder script: 100%  6.27k/6.27k [00:00<00:00, 446kB/s]
```

```
print(rouge_base_med_dialog)
print(rouge_i)
```

```
{'rouge1': np.float64(0.20294088749028538), 'rouge2': np.float64(0.02361764621026269), 'rougeL': np.float64(0.10506680040438876), 'r
{'rouge1': np.float64(0.38513670670116207), 'rouge2': np.float64(0.288323220950919), 'rougeL': np.float64(0.3663538906069719), 'roug
```

```
# plot rouge scores
rouge_dict = {
    'rouge1': (rouge_base_med_dialog['rouge1'], rouge_i['rouge1']),
    'rouge2': (rouge_base_med_dialog['rouge2'], rouge_i['rouge2']),
    'rougeL': (rouge_base_med_dialog['rougeL'], rouge_i['rougeL']),
    'rougeLsum': (rouge_base_med_dialog['rougeLsum'], rouge_i['rougeLsum'])
}
model_labels = ('Base', 'Model I')

x = np.arange(len(model_labels))
```

```
x = np.arange(len(model_labels))
width = 0.2
multiplier = 0

fig, ax = plt.subplots(layout='constrained')

for attribute, measurement in rouge_dict.items():
    offset = width * multiplier
    msr = np.round(measurement, 4) # round to 4 decimal places for plot
    rects = ax.bar(x + offset, msr, width, label=attribute)
    ax.bar_label(rects, padding=1)
    multiplier += 1

ax.set_ylabel('Rouge Score')
ax.set_title('Rouge Score Comparison')
ax.set_xticks(x + width, model_labels)
ax.legend(loc='upper left', ncols=4)
ax.set_ylim(0, 1)

plt.show()
```




```
bleu_base_med_dialog = calc_bleu_metric(qa_references, base_med_dialog_response_list)
bleu_i = calc_bleu_metric(qa_references, i_response_list)
```

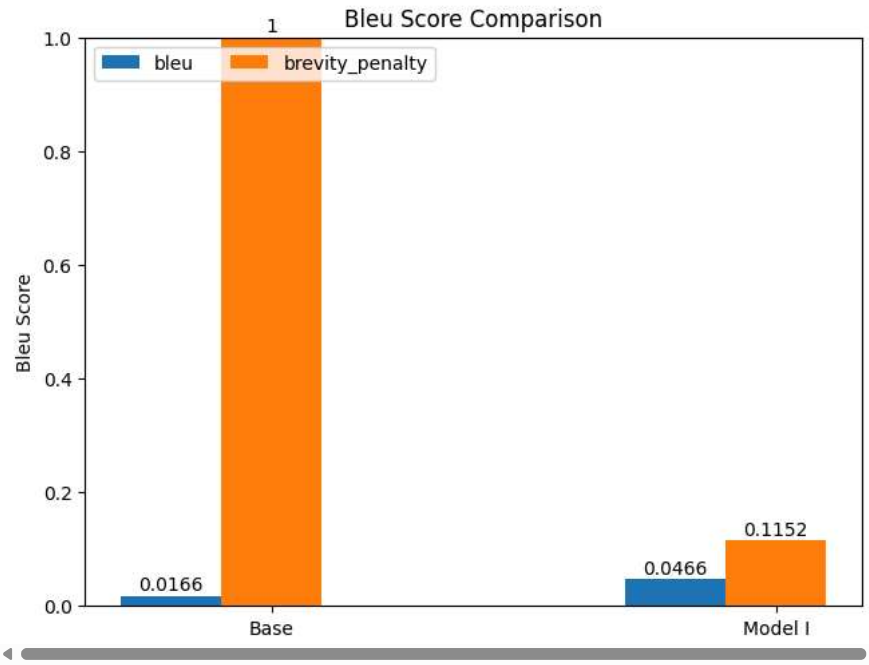
```
Downloading builder script: 100%  5.94k/5.94k [00:00<00:00, 687kB/s]
Downloading extra modules:  4.07k/? [00:00<00:00, 465kB/s]
Downloading extra modules: 100%  3.34k/3.34k [00:00<00:00, 376kB/s]
```

```
print(bleu_base_med_dialog)
print(bleu_i)
```

```
: 0.016551047075983986, 'precisions': [0.21872322193658955, 0.0323066982554383, 0.005738414898224339, 0.001850642281733072], 'brevit
: 0.04662877916583039, 'precisions': [0.588534442903375, 0.4103170847136772, 0.34997576345128456, 0.31793343268753105], 'brevity_pen
```

```
# plot bleu scores
bleu_dict = {
    'bleu': (bleu_base_med_dialog['bleu'], bleu_i['bleu']),
    'brevity_penalty': (bleu_base_med_dialog['brevity_penalty'], bleu_i['brevity_penalty'])
}

model_labels = ('Base', 'Model I')

x = np.arange(len(model_labels))
width = 0.2
multiplier = 0

fig, ax = plt.subplots(layout='constrained')

for attribute, measurement in bleu_dict.items():
    offset = width * multiplier
    msr = np.round(measurement, 4) # round to 4 decimal places for plot
```

```
    rects = ax.bar(x + offset, msr, width, label=attribute)
    ax.bar_label(rects, padding=1)
    multiplier += 1

ax.set_ylabel('Bleu Score')
ax.set_title('Bleu Score Comparison')
ax.set_xticks(x + width, model_labels)
ax.legend(loc='upper left', ncols=4)
ax.set_ylim(0, 1)

plt.show()
```




```
meteor_base_med_dialog = calc_meteor_metric(qa_references, base_med_dialog_response_list)
meteor_i = calc_meteor_metric(qa_references, i_response_list)
```

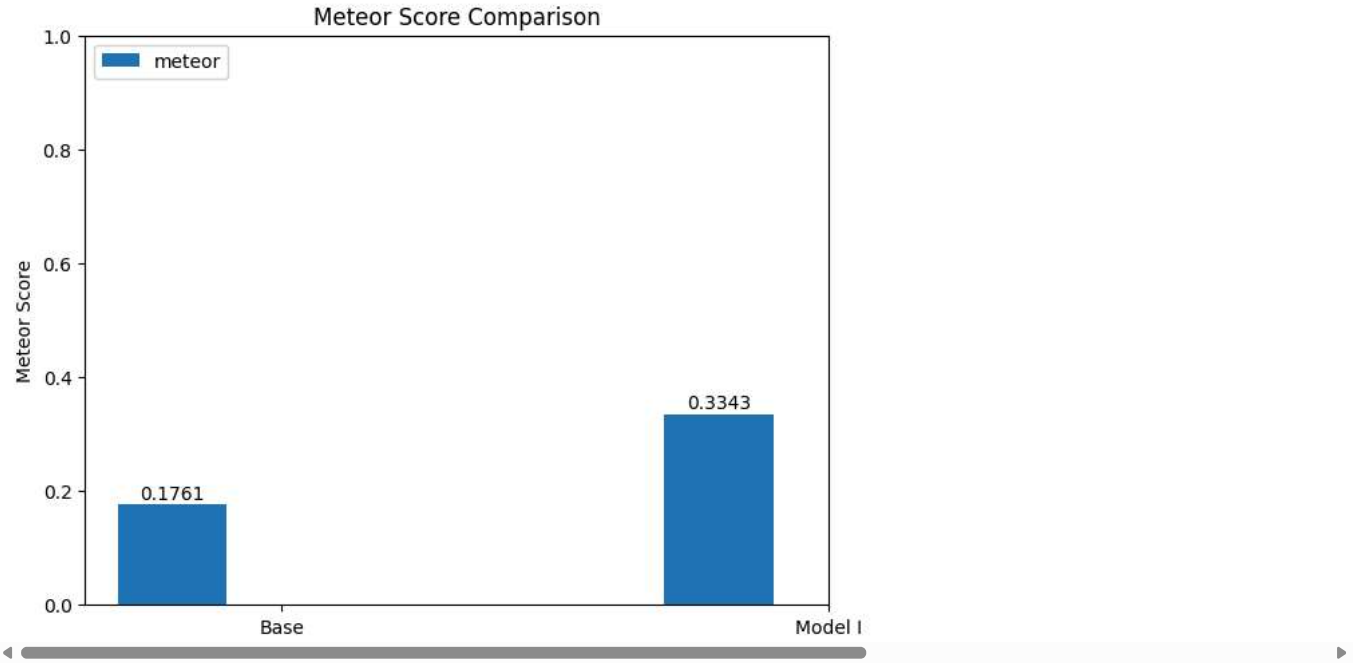
```
Downloading builder script: 100%  7.02k/7.02k [00:00<00:00, 774kB/s]
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
```

```
print(meteor_base_med_dialog)
print(meteor_i)
```

```
{'meteor': np.float64(0.17605032981349583)}
{'meteor': np.float64(0.3343370780511959)}
```

```
# plot meteor scores
meteor_dict = {
    'meteor': (meteor_base_med_dialog['meteor'], meteor_i['meteor'])
}
model_labels = ('Base', 'Model I')

x = np.arange(len(model_labels))
width = 0.2
multiplier = 0

fig, ax = plt.subplots(layout='constrained')

for attribute, measurement in meteor_dict.items():
    offset = width * multiplier
    msr = np.round(measurement, 4) # round to 4 decimal places for plot
    rects = ax.bar(x + offset, msr, width, label=attribute)
    ax.bar_label(rects, padding=1)
    multiplier += 1

ax.set_ylabel('Meteor Score')
ax.set_title('Meteor Score Comparison')
```

```
ax.set_xticks(x + width, model_labels)
ax.legend(loc='upper left', ncols=4)
ax.set_ylim(0, 1)

plt.show()
```




```
bert_score_base_med_dialog = calc_bert_score(qa_references, base_med_dialog_response_list)
bert_score_i = calc_bert_score(qa_references, i_response_list)
```

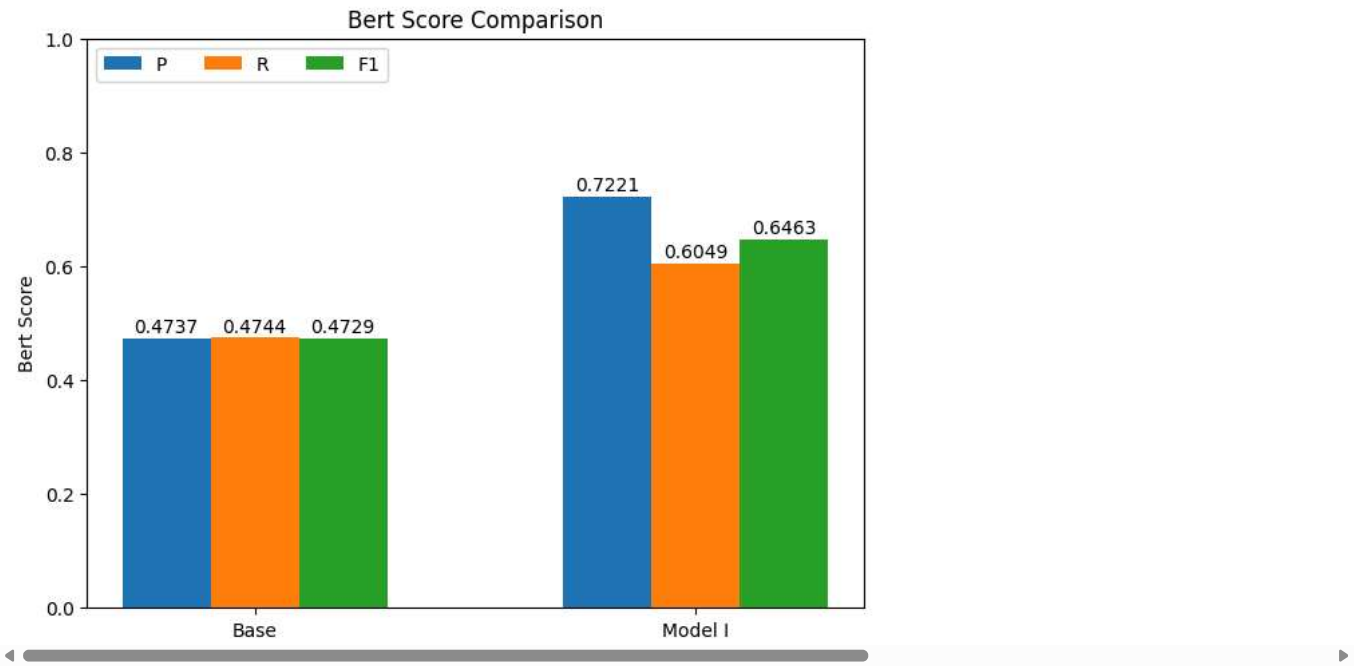
```
tokenizer_config.json: 100%  48.0/48.0 [00:00<00:00, 6.16kB/s]
config.json: 100%  570/570 [00:00<00:00, 39.5kB/s]
vocab.txt: 100%  232k/232k [00:00<00:00, 546kB/s]
tokenizer.json: 100%  466k/466k [00:00<00:00, 2.17MB/s]
Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better p
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back
model.safetensors: 100%  440M/440M [00:01<00:00, 304MB/s]
```

```
print(bert_score_base_med_dialog)
print(bert_score_i)
```

```
{'P': np.float32(0.47369266), 'R': np.float32(0.474428), 'F1': np.float32(0.47294253)}
{'P': np.float32(0.72214913), 'R': np.float32(0.6048881), 'F1': np.float32(0.6463166)}
```

```
# plot bert scores
bert_score_dict = {
    'P': (bert_score_base_med_dialog['P'], bert_score_i['P']),
    'R': (bert_score_base_med_dialog['R'], bert_score_i['R']),
    'F1': (bert_score_base_med_dialog['F1'], bert_score_i['F1'])
}
model_labels = ('Base', 'Model I')

x = np.arange(len(model_labels))
width = 0.2
multiplier = 0

fig, ax = plt.subplots(layout='constrained')

for attribute, measurement in bert_score_dict.items():
    offset = width * multiplier
    msr = np.round(measurement, 4) # round to 4 decimal places for plot
    rects = ax.bar(x + offset, msr, width, label=attribute)
    ax.bar_label(rects, padding=1)
    multiplier += 1

ax.set_ylabel('Bert Score')
ax.set_title('Bert Score Comparison')
ax.set_xticks(x + width, model_labels)
ax.legend(loc='upper left', ncols=4)
ax.set_ylim(0, 1)

```

```
plt.show()
```




```
# human evaluation
def print_response(QA_NUM):
  for i in range(QA_NUM):
    print('----------------------------------------')
    print('\nTest question: ' + qa_list[i][0])
    print('\nTest answer: ' + qa_list[i][1])
    print('\nBase: ' + base_med_dialog_response_list[i])
    print('\nModel I: ' + i_response_list[i])

print_response(5)
```

```
----------------------------------------

Test question: Hi doctor, I am a 26 year old male. I am 5 feet and 9 inches tall and weigh 255 pounds. When I eat spicy food, I pc

Test answer: Hello. I have gone through your information and test reports (attachment removed to protect patient identity). So, ir

Base:  Thank you for reaching out to me with your concerns. I understand that you are experiencing some unusual symptoms, includir

Based on the information you have provided, I would recommend that you schedule an appointment with your primary care physician or

In the meantime, here are some things you can do to help manage your symptoms:

1. Avoid spicy foods: If you notice that spicy foods trigger your symptoms, try to avoid them for a while to see if your symptoms
2. Stay hydrated: Drinking plenty of water can help to soften your stool and make it easier to pass.
3. Consider probiotics: Probiotics are live bacteria and ye

Model I: Hello. I have gone through your information and reports (attachment removed to protect patient identity). For further dou
----------------------------------------

Test question: Hi doctor, I am not sure exactly when I started having a split stream urination problem, but I really worried about

Test answer: Hi. Revert back with the reports to a urologist online --> https://www.icliniq.com/ask-a-doctor-online/urologist

Base:  Thank you for reaching out for assistance. It's completely normal to experience some degree of split stream urination, espe

The dribbling you experience after urination is a common symptom of an overactive bladder, which can be caused by a variety of fac

In the meantime, there are some things you can try to help manage your symptoms:

1. Practice bladder training: This involves gradually increasing the amount of time between trips to the bathroom to help strengtl
2. Avoid triggers: Identify and avoid any triggers that may be causing your symptoms, such as caffeine or spicy foods.
3. Use the right position: When ur

Model I: Hi. I have gone through your query and understand your concerns. For further information consult a urologist online --> h
----------------------------------------

Test question: Hi doctor, I am a 46 year old female, and I am obese. I do not drink caffeinated drinks, smoke, or do drugs. I took

Test answer: Hi. For more information consult a cardiologist online --> https://www.icliniq.com/ask-a-doctor-online/cardiologist

Base:  Thank you for reaching out for medical assistance. I understand your concerns about your recent heart palpitations and the

Based on the information you've provided, it's important to consult with a cardiologist or a cardiovascular specialist to further

Given your history of obesity, high blood pressure, and previous cardiac issues, it's possible that these factors may be contribut
```

```
In the meantime, here are some suggestions that may help alleviate your symptoms:

1. Keep a heart healthy diary: Track your food intake,

Model I: Hi. I have gone through your query and understand your concerns. For further information consult a cardiologist online --
----------------------------------------

Test question: Hello doctor,I was positive for hepatitis B four years ago, and recently I was rushed to the hospital for coughing

Test answer: Hello. As your gene Xpert report (attachment removed to protect patient identity) shows you have been tested positive
```

7. Evaluation Summary

Changing the dataset to MedDialog IC shows that the fine-tuned Model_I performs better than the base model.

On human inspection the Llama 2 base model provides a more conversational response, however the fine-tuned model response aligns better with the dataset, indicating that it has learned from the training.

This suggests that the base Llama model was not trained on MedDialog dataset, whereas PubMedQA dataset may have been used to train Llama 2.

Next steps:

- Continue to experiment with further hyperparameter tuning
- Increase volume of training data
- Adjust system prompt, and model parameters on inference
- Improve the pre-processing of the dataset, to provide a more useful answer via the Chatbot

---

```
#install necessary packages for printing to pdf
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
!pip install pypandoc
!pip install nbconvert
```

```
/sbin/ldconfig.real: /usr/local/lib/libur_adapter_opencl.so.0 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc_proxy.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libhwloc.so.15 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link
```