First Full LoRA Trial with Transformer

peft (for LoRA) and FLAN-T5-small for the LLM

I'm following what seems to be a great tutorial from Mehul Gupta,

> https://medium.com/data-science-in-your-pocket/lora-for-fine-tuning-llms-explained-with-codes-and-example-62a7ac5a3578
>
> https://web.archive.org/web/20240522140323/https://medium.com/data-science-in-your-pocket/lora-for-fine-tuning-llms-explained-with-codes-and-example-62a7ac5a3578

I'm doing this to prepare creating a LoRA for RWKV (@todo put links in here) so as to fine-tune it for Pat's OLECT-LM stuff.

In [1]:
```
# #  No need to run this again
# !powershell -c (Get-Date -UFormat \"%s_%Y%m%dT%H%M%S%Z00\") -replace '[.][0-9]*_', '_'"
```

Output was:

```
1716367147_20240522T083907-0600
```

Imports

In [43]:
```
from datasets import load_dataset
from random import randrange
import torch
from transformers import AutoTokenizer, \
                         AutoModelForSeq2SeqLM, \
                         TrainingArguments, \
                         pipeline
from transformers.utils import logging
from peft import LoraConfig, \
                 prepare_model_for_kbit_training, \
                 get_peft_model, \
```

```
                  AutoPeftModelForCausalLM
from trl import SFTTrainer
from huggingface_hub import login, notebook_login

from datasets import load_metric
import nltk
import rouge_score

import pickle
import pprint
import timeit
from humanfriendly import format_timespan
import os
```

Load the training and test dataset along with the LLM with its tokenizer

The LLM will be fine-tuned. It seems the tokenizer will also be fine-tuned, but I'm not sure

Why aren't we loading the validation set? (I don't know; that's not a teaching question.)

I've tried to make use of it with the `trainer`. We'll see how it goes

In [35]:
```
#  Need to install  datasets  from pip, not conda. I'll do all from pip.
#+ I'll get rid of the current conda environment and make it anew.
#+ Actually, I'll make sure  conda  and  pip  are updated, then do what
#+ I discussed above.
#+
#+ cf.
#+     arch_ref_1 = "https://web.archive.org/web/20240522150357/" + \
#+                  "https://stackoverflow.com/questions/77433096/" + \
#+                  "notimplementederror-loading-a-dataset-" + \
#+                  "cached-in-a-localfilesystem-is-not-suppor"
#+
#+ Also useful might be
#+     arch_ref_2 = "https://web.archive.org/web/20240522150310/" + \
#+                  "https://stackoverflow.com/questions/76340743/" + \
#+                  "huggingface-load-datasets-gives-" + \
#+                  "notimplementederror-cannot-error"
#
data_files = {'train':'samsum-train.json',
```

```
              'evaluation':'samsum-validation.json',
              'test':'samsum-test.json'}
dataset = load_dataset('json', data_files=data_files)

model_name = "google/flan-t5-small"
model = AutoModelForSeq2SeqLM.from_pretrained(model_name)

#  Next line makes training faster but a little less accurate
model.config.pretraining_tp = 1

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

#  padding instructions for the tokenizer
#+   ??? !!! What about for RWKV !!! ???
tokenizer.pad_token = tokenizer.eos_token
tokenizer.padding_side = "right"
```

```
Generating train split: 0 examples [00:00, ? examples/s]
Generating evaluation split: 0 examples [00:00, ? examples/s]
Generating test split: 0 examples [00:00, ? examples/s]
```

Trying some things I've been learning

In [5]:
```
print(model)
```

```
T5ForConditionalGeneration(
  (shared): Embedding(32128, 512)
  (encoder): T5Stack(
    (embed_tokens): Embedding(32128, 512)
    (block): ModuleList(
      (0): T5Block(
        (layer): ModuleList(
          (0): T5LayerSelfAttention(
            (SelfAttention): T5Attention(
              (q): Linear(in_features=512, out_features=384, bias=False)
              (k): Linear(in_features=512, out_features=384, bias=False)
              (v): Linear(in_features=512, out_features=384, bias=False)
              (o): Linear(in_features=384, out_features=512, bias=False)
              (relative_attention_bias): Embedding(32, 6)
            )
            (layer_norm): T5LayerNorm()
            (dropout): Dropout(p=0.1, inplace=False)
          )
          (1): T5LayerFF(
            (DenseReluDense): T5DenseGatedActDense(
              (wi_0): Linear(in_features=512, out_features=1024, bias=False)
              (wi_1): Linear(in_features=512, out_features=1024, bias=False)
              (wo): Linear(in_features=1024, out_features=512, bias=False)
              (dropout): Dropout(p=0.1, inplace=False)
              (act): NewGELUActivation()
            )
            (layer_norm): T5LayerNorm()
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
      )
      (1-7): 7 x T5Block(
        (layer): ModuleList(
          (0): T5LayerSelfAttention(
            (SelfAttention): T5Attention(
              (q): Linear(in_features=512, out_features=384, bias=False)
              (k): Linear(in_features=512, out_features=384, bias=False)
              (v): Linear(in_features=512, out_features=384, bias=False)
              (o): Linear(in_features=384, out_features=512, bias=False)
            )
            (layer_norm): T5LayerNorm()
            (dropout): Dropout(p=0.1, inplace=False)
```

```
          )
          (1): T5LayerFF(
            (DenseReluDense): T5DenseGatedActDense(
              (wi_0): Linear(in_features=512, out_features=1024, bias=False)
              (wi_1): Linear(in_features=512, out_features=1024, bias=False)
              (wo): Linear(in_features=1024, out_features=512, bias=False)
              (dropout): Dropout(p=0.1, inplace=False)
              (act): NewGELUActivation()
            )
            (layer_norm): T5LayerNorm()
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
      )
    )
    (final_layer_norm): T5LayerNorm()
    (dropout): Dropout(p=0.1, inplace=False)
  )
  (decoder): T5Stack(
    (embed_tokens): Embedding(32128, 512)
    (block): ModuleList(
      (0): T5Block(
        (layer): ModuleList(
          (0): T5LayerSelfAttention(
            (SelfAttention): T5Attention(
              (q): Linear(in_features=512, out_features=384, bias=False)
              (k): Linear(in_features=512, out_features=384, bias=False)
              (v): Linear(in_features=512, out_features=384, bias=False)
              (o): Linear(in_features=384, out_features=512, bias=False)
              (relative_attention_bias): Embedding(32, 6)
            )
            (layer_norm): T5LayerNorm()
            (dropout): Dropout(p=0.1, inplace=False)
          )
          (1): T5LayerCrossAttention(
            (EncDecAttention): T5Attention(
              (q): Linear(in_features=512, out_features=384, bias=False)
              (k): Linear(in_features=512, out_features=384, bias=False)
              (v): Linear(in_features=512, out_features=384, bias=False)
              (o): Linear(in_features=384, out_features=512, bias=False)
            )
            (layer_norm): T5LayerNorm()
```

```
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (2): T5LayerFF(
      (DenseReluDense): T5DenseGatedActDense(
        (wi_0): Linear(in_features=512, out_features=1024, bias=False)
        (wi_1): Linear(in_features=512, out_features=1024, bias=False)
        (wo): Linear(in_features=1024, out_features=512, bias=False)
        (dropout): Dropout(p=0.1, inplace=False)
        (act): NewGELUActivation()
      )
      (layer_norm): T5LayerNorm()
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
)
(1-7): 7 x T5Block(
  (layer): ModuleList(
    (0): T5LayerSelfAttention(
      (SelfAttention): T5Attention(
        (q): Linear(in_features=512, out_features=384, bias=False)
        (k): Linear(in_features=512, out_features=384, bias=False)
        (v): Linear(in_features=512, out_features=384, bias=False)
        (o): Linear(in_features=384, out_features=512, bias=False)
      )
      (layer_norm): T5LayerNorm()
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (1): T5LayerCrossAttention(
      (EncDecAttention): T5Attention(
        (q): Linear(in_features=512, out_features=384, bias=False)
        (k): Linear(in_features=512, out_features=384, bias=False)
        (v): Linear(in_features=512, out_features=384, bias=False)
        (o): Linear(in_features=384, out_features=512, bias=False)
      )
      (layer_norm): T5LayerNorm()
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (2): T5LayerFF(
      (DenseReluDense): T5DenseGatedActDense(
        (wi_0): Linear(in_features=512, out_features=1024, bias=False)
        (wi_1): Linear(in_features=512, out_features=1024, bias=False)
        (wo): Linear(in_features=1024, out_features=512, bias=False)
```

```
            (dropout): Dropout(p=0.1, inplace=False)
            (act): NewGELUActivation()
          )
          (layer_norm): T5LayerNorm()
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
    )
  )
  (final_layer_norm): T5LayerNorm()
  (dropout): Dropout(p=0.1, inplace=False)
 )
 (lm_head): Linear(in_features=512, out_features=32128, bias=False)
)
```

Prompt and Trainer

For our SFT (**S**upervised **F**ine **T**uning) model, we use the `class trl.SFTTrainer`.

I want to research this a bit, especially the `formatting_func` that we'll be passing to the `SFTTrainer`.

First, though, some information about SFT. From the Hugging Face Documentation at https://huggingface.co/docs/trl/en/sft_trainer (archived)

> Supervised fine-tuning (or SFT for short) is a crucial step in RLHF. In TRL we provide an easy-to-use API to create your SFT models and train them with few lines of code on your dataset.

Though I won't be using the examples unless I get even more stuck, the next paragraph *has* examples, and I'll put the paragraph here.

> Check out a complete flexible example at examples/scripts/sft.py [archived]. Experimental support for Vision Language Models is also included in the example examples/scripts/vsft_llava.py [archived].

RLHF (archived wikipedia page) is **R**einforcement **L**earning from **H**uman **F**eedback. TRL%20step.) (archived) **T**ransfer **R**einforcement **L**earning, a library from Hugging Face.

For the parameter, `formatting_func`, I can look ath the documentation site above (specifically here), at the GitHub repo for the code (in the docstrings), or from my local `conda` environment, at `C:\Users\bballdave025\.conda\envs\rwkv-lora-`

`pat\Lib\site-packages\trl\trainer\sft_trainer.py`.

Pulling code from the last one, I get

```
    formatting_func (`Optional[Callable]`):
       The formatting function to be used for creating the `ConstantLengthDataset`.
```

That matches the first very well

> **formatting_func** (`Optional[Callable]`) — The formatting function to be used for creating the `ConstantLengthDataset`.

(A quick note: In this Jupyter Notebook environment, I could have typed `trainer = SFTTrainer(` and then `Shift` + `Tab` to find that same documentation.

However, I think that more clarity is found at the documentation for `ConstantLengthDataset

> **formatting_func** (`Callable`, **optional**) — Function that formats the text before tokenization. Usually it is recommended to have follows a certain pattern such as `"### Question: {question} ### Answer: {answer}"`

So, as we'll see the next code from the tutorial, it basically is a prompt templater/formatter that matches the JSON. For example, we use `sample['dialogue']` to access the `dialogue` key/pair. That's what I got from all this stuff.

Mehul Gupta himself stated

> Next, using the Input and Output, we will create a prompt template which is a requirement by the SFTTrainer we will be using later

Prompt

In [6]:
```
def prompt_instruction_format(sample):
    return f""" Instruction:
      Use the Task below and the Input given to write the Response:

      ### Task:
      Summarize the Input
```

```
      ### Input:
      {sample['dialogue']}

      ### Response:
      {sample['summary']}
      """
##endof:  prompt_instruction_format(sample)
```

Trainer - the LoRA Setup Part

Arguments and Configuration

In [7]:
```
#  Some arguments to pass to the trainer
training_args = TrainingArguments( output_dir='output',
                                   num_train_epochs=1,
                                   per_device_train_batch_size=4,
                                   save_strategy='epoch',
                                   learning_rate=2e-4
)

# the fine-tuning (peft for LoRA) stuff
peft_config = LoraConfig( lora_alpha=16,
                          lora_dropout=0.1,
                          r=64,
                          bias='none',
                          task_type='CAUSAL_LM'
)
```

`task_type` , cf. https://github.com/huggingface/peft/blob/main/src/peft/config.py#L222

```
    Args:
        peft_type (Union[[`~peft.utils.config.PeftType`], `str`]): The type of Peft method to
use.
        task_type (Union[[`~peft.utils.config.TaskType`], `str`]): The type of task to perform.
        inference_mode (`bool`, defaults to `False`): Whether to use the Peft model in
inference mode.
```

After some searching using Cygwin

```
bballdave025@MYMACHINE /cygdrive/c/Users/bballdave025/.conda/envs/rwkv-lora-pat/Lib/site-
packages/peft/utils
$ ls -lah
total 116K
drwx------+ 1 bballdave025 bballdave025    0 May 28 21:09 .
drwx------+ 1 bballdave025 bballdave025    0 May 28 21:09 ..
-rwx------+ 1 bballdave025 bballdave025 2.0K May 28 21:09 __init__.py
drwx------+ 1 bballdave025 bballdave025    0 May 28 21:09 __pycache__
-rwx------+ 1 bballdave025 bballdave025 8.0K May 28 21:09 constants.py
-rwx------+ 1 bballdave025 bballdave025 3.8K May 28 21:09 integrations.py
-rwx------+ 1 bballdave025 bballdave025  17K May 28 21:09 loftq_utils.py
-rwx------+ 1 bballdave025 bballdave025 9.7K May 28 21:09 merge_utils.py
-rwx------+ 1 bballdave025 bballdave025  25K May 28 21:09 other.py
-rwx------+ 1 bballdave025 bballdave025 2.2K May 28 21:09 peft_types.py
-rwx------+ 1 bballdave025 bballdave025  21K May 28 21:09 save_and_load.py

bballdave025@MYMACHINE /cygdrive/c/Users/bballdave025/.conda/envs/rwkv-lora-pat/Lib/site-
packages/peft/utils
$ grep -iIRHn "TaskType" .
peft_types.py:60:class TaskType(str, enum.Enum):
__init__.py:20:# from .config import PeftConfig, PeftType, PromptLearningConfig, TaskType
__init__.py:22:from .peft_types import PeftType, TaskType

bballdave025@MYMACHINE /cygdrive/c/Users/bballdave025/.conda/envs/rwkv-lora-pat/Lib/site-
packages/peft/utils
$
```

So, let's look at the `peft_types.py` file.

The docstring for `class TaskType(str, enum.Enum)` is

```
    Enum class for the different types of tasks supported by PEFT.

    Overview of the supported task types:
    - SEQ_CLS: Text classification.
    - SEQ_2_SEQ_LM: Sequence-to-sequence language modeling.
    - CAUSAL_LM: Causal language modeling.
    - TOKEN_CLS: Token classification.
```

```
    - QUESTION_ANS: Question answering.
    - FEATURE_EXTRACTION: Feature extraction. Provides the hidden states which can be used as
embeddings or features
      for downstream tasks.
```

We're going to start timing stuff, so here's some system info

`win_system_info_as_script.py` is a script I wrote with the help of a variety of StackOverflow and documentation sources. It should be in the working directory.

In [44]:
```
import win_system_info_as_script as winsysinfo
winsysinfo.run()
```

```
##########################  System Information  ##########################
System: Windows
Node Name: NOT-FOR-NOW
Release: 10
Version: 10.0.19045
Machine: AMD64
Processor: Intel64 Family 6 Model 165 Stepping 3, GenuineIntel
Processor: Intel(R) Core(TM) i3-10100 CPU @ 3.60GHz
Ip-Address: NOT-FOR-NOW
Mac-Address: NOT-FOR-NOW

############################  Boot Time  ############################
Boot Time (date and time of last boot) was
Boot Time: 2024-5-26T14:29:1

############################  CPU Info  ############################
Physical cores: 4
Total cores: 8
CPU Usage Per Core:
Core 0: 4.7%
Core 1: 3.1%
Core 2: 6.2%
Core 3: 9.4%
Core 4: 4.7%
Core 5: 3.1%
Core 6: 6.2%
Core 7: 3.1%
Total CPU Usage: 26.7%
Max Frequency: 3600.00Mhz
Min Frequency: 0.00Mhz
Current Frequency: 3600.00Mhz

############################  GPU Info  ############################
Information on GPU(s)/Graphics Card(s)
 (if any such information is to be found)

Using  wmi , we get the following  win32_VideoController  names.
   Trigger 6 External Graphics
   Intel(R) UHD Graphics 630
Using  PyTorch  and the  torch.cuda.is_available()  method.
The statement, 'There is CUDA and an appropriate GPU',
  is ...  False
```

```
Using  TensorFlow  with several of its methods.
  Attempting to get GPU Device List
No GPU Devices.
Tensorflow can give us CPU (and/or GPU) info.
The info here might help you know if we're running on a CPU.

Trying to use some nvidia code ( nvidia-smi ) to find information
  That's the end of the nvidia try.

Those are all our chances to find out about any GPU/Graphics Cards.

######################  Memory (RAM) Information  ######################
Total: 31.67GbB
Available: 16.54GbB
Used: 15.13GbB
Percentage: 47.8%
      =============== SWAP Memory ===============
Total: 4.75GbB
Free: 4.41GbB
Used: 352.62MbB
Percentage: 7.2%

##############################  Disk Info  ##############################
Partitions and Usage:
=== Device: C:\ ===
  Mountpoint: C:\
  File system type: NTFS
  Total Size: 915.94GbB
  Used: 587.17GbB
  Free: 328.78GbB
  Percentage: 64.1%
=== Device: D:\ ===
  Mountpoint: D:\
  File system type: exFAT
  Total Size: 12.73TbB
  Used: 1.99TbB
  Free: 10.75TbB
  Percentage: 15.6%
=== Device: E:\ ===
  Mountpoint: E:\
  File system type: FAT32
  Total Size: 115.31GbB
```

```
  Used: 46.08GbB
  Free: 69.23GbB
  Percentage: 40.0%
  Since last boot,
Total read: 157.49GbB
Total write: 198.10GbB
```

```
  That nvidia stuff didn't work
  The error information is:
[WinError 2] The system cannot find the file specified
```

Try for a baseline

Just one summarization to begin with, randomly picked

In [1]:
```
!powershell -c (Get-Date -UFormat \"%s_%Y%m%dT%H%M%S%Z00\") -replace '[.][0-9]*_', '_'"
```

Output was:

```
timestamp
```

In [8]:
```
#  Just one summarization to begin with, randomly picked ... but
#+ now with th possibility of a known seed, to allow visual
#+ comparison with after-training results.
#+ I'M NOT GOING TO USE THIS REPEATED SEED, I'm just going to
#+ use the datum at the first index to compare.

do_seed_for_repeatable = False

summarizer = pipeline('summarization', model=model, tokenizer=tokenizer)

if do_seed_for_repeatable:
  rand_seed_for_randrange = 137
  random.seed(rand_seed_for_randrange)
##endof:  if do_seed_for_repeatable

sample = dataset['test'][randrange(len(dataset["test"]))]
print(f"dialogue: \n{sample['dialogue']}\n---------------")
```

```
res = summarizer(sample["dialogue"])

print(f"flan-t5-small summary:\n{res[0]['summary_text']}")
```

```
Your max_length is set to 200, but your input_length is only 137. Since this is a summarization task, where outputs s
horter than the input are typically wanted, you might consider decreasing max_length manually, e.g. summarizer('...',
max_length=68)
dialogue:
Sam: Where are you?
Tilly: Just leaving school now?
Sam: What have you been doing? You said you would be home for 4!
Tilly: Yes had to go for detention
Sam: I asked your mum if that's what had happened but she said she hadn't got a text
Tilly: They are useless. They are supposed to tell your parents if you get a detention.
Sam: How long will you be?
Tilly: About 40 minutes
Sam: OK.. I'm going back to mine - phone me when you are home.
Tilly: OK - see you soon.
---------------
flan-t5-small summary:
Sam is leaving school now. Tilly will be home for about 40 minutes. Sam will call Tilly when he is home. Sam is going
back to his place.
```

Now, one summarization with comparison to ground truth

In [17]:
```
# Now one summarization with comparison to ground truth

summarizer = pipeline('summarization', model=model, tokenizer=tokenizer)

pred_test_list = []
ref_test_list = []

sample_num = 0

this_sample = dataset['test'][sample_num]

print(f"dialogue: \n{this_sample['dialogue']}\n---------------")

grnd_summary = this_sample['summary']
res = summarizer(this_sample['dialogue'])
res_summary = res[0]['summary_text']
```

```
# humgen is for human-generated

print(f"human-genratd summary:\n{grnd_summary}")
print(f"flan-t5-small summary:\n{res_summary}")

ref_test_list.append(grnd_summary)
pred_test_list.append(res_summary)

print("\n\n---------- ROUGE SCORES ----------")

rouge = load_metric('rouge', trust_remote_code=True)

results = rouge.compute(predictions=pred_test_list,
                        references=ref_test_list,
                        use_aggregator=True)

# >>> print(list(results.keys()))
# ['rouge1', 'rouge2', 'rougeL', 'rougeLsum']

print()
print("ROUGE-1 results")
pprint.pp(results['rouge1'])
print()
print("ROUGE-2 results")
pprint.pp(results['rouge2'])
print()
print("ROUGE-L results")
pprint.pp(results['rougeL'])
print()
print("ROUGE-Lsum results")
pprint.pp(results['rougeLsum'])
```

```
Your max_length is set to 200, but your input_length is only 133. Since this is a summarization task, where outputs s
horter than the input are typically wanted, you might consider decreasing max_length manually, e.g. summarizer('...',
max_length=66)
```

```
dialogue:
Hannah: Hey, do you have Betty's number?
Amanda: Lemme check
Hannah: <file_gif>
Amanda: Sorry, can't find it.
Amanda: Ask Larry
Amanda: He called her last time we were at the park together
Hannah: I don't know him well
Hannah: <file_gif>
Amanda: Don't be shy, he's very nice
Hannah: If you say so..
Hannah: I'd rather you texted him
Amanda: Just text him 🙂
Hannah: Urgh.. Alright
Hannah: Bye
Amanda: Bye bye
---------------
human-genratd summary:
Hannah needs Betty's number but Amanda doesn't have it. She needs to contact Larry.
flan-t5-small summary:
Larry called Hannah last time she was at the park together. Hannah doesn't know Larry well. Larry called her last tim
e they were at a park. Hannah will text Larry.


---------- ROUGE SCORES ----------

ROUGE-1 results
AggregateScore(low=Score(precision=0.16129032258064516, recall=0.3125, fmeasure=0.2127659574468085), mid=Score(precis
ion=0.16129032258064516, recall=0.3125, fmeasure=0.2127659574468085), high=Score(precision=0.16129032258064516, recal
l=0.3125, fmeasure=0.2127659574468085))

ROUGE-2 results
AggregateScore(low=Score(precision=0.03333333333333333, recall=0.06666666666666667, fmeasure=0.04444444444444444), mi
d=Score(precision=0.03333333333333333, recall=0.06666666666666667, fmeasure=0.04444444444444444), high=Score(precisio
n=0.03333333333333333, recall=0.06666666666666667, fmeasure=0.04444444444444444))

ROUGE-L results
AggregateScore(low=Score(precision=0.12903225806451613, recall=0.25, fmeasure=0.1702127659574468), mid=Score(precisio
n=0.12903225806451613, recall=0.25, fmeasure=0.1702127659574468), high=Score(precision=0.12903225806451613, recall=0.
25, fmeasure=0.1702127659574468))

ROUGE-Lsum results
```

```
AggregateScore(low=Score(precision=0.12903225806451613, recall=0.25, fmeasure=0.1702127659574468), mid=Score(precisio
n=0.12903225806451613, recall=0.25, fmeasure=0.1702127659574468), high=Score(precision=0.12903225806451613, recall=0.
25, fmeasure=0.1702127659574468))
```

Verbosity stuff - get rid of the nice advice

In [ ]:
```
!powershell -c (Get-Date -UFormat \"%s_%Y-%m-%dT%H%M%S%Z00\") -replace '[.][0-9]*_', '_'
```

Output was:

```
timestamp
```

In [30]:
```
# bballdave025@MYMACHINE /cygdrive/c/Users/bballdave025/.conda/envs/rwkv-lora-pat/Lib/site-packages/peft/utils
# $ date +'%s_%Y-%m-%dT%H%M%S%z'
# 1717049876_2024-05-30T001756-0600

log_verbosity_is_critical = \
  logging.get_verbosity() == logging.CRITICAL # alias FATAL, 50
log_verbosity_is_error = \
  logging.get_verbosity() == logging.ERROR # 40
log_verbosity_is_warn = \
  logging.get_verbosity() == logging.WARNING # alias WARN, 30
log_verbosity_is_info = \
  logging.get_verbosity() == logging.INFO # 20
log_verbosity_is_debug = \
  logging.get_verbosity() == logging.DEBUG # 10

print( "The statement, 'logging verbosity is CRITICAL' " + \
      f"is {log_verbosity_is_critical}")
print( "The statement, 'logging verbosity is    ERROR' " + \
      f"is {log_verbosity_is_error}")
print( "The statement, 'logging verbosity is  WARNING' " + \
      f"is {log_verbosity_is_warn}")
print( "The statement, 'logging verbosity is     INFO' " + \
      f"is {log_verbosity_is_info}")
print( "The statement, 'logging verbosity is    DEBUG' " + \
      f"is {log_verbosity_is_debug}")

print()

init_log_verbosity = logging.get_verbosity()
```

```
print(f"The value of logging.get_verbosity() is: {init_log_verbosity}")

print()

init_t_n_a_w = os.environ.get('TRANSFORMERS_NO_ADVISORY_WARNINGS')
print(f"TRANSFORMERS_NO_ADIVSORY_WARNINGS: {init_t_n_a_w}")
```

```
The statement, 'logging verbosity is CRITICAL' is False
The statement, 'logging verbosity is    ERROR' is False
The statement, 'logging verbosity is  WARNING' is True
The statement, 'logging verbosity is     INFO' is False
The statement, 'logging verbosity is    DEBUG' is False

The value of logging.get_verbosity() is: 30

TRANSFORMERS_NO_ADIVSORY_WARNINGS: None
```

Actual Baseline

In [ ]:
```
!powershell -c (Get-Date -UFormat \"%s_%Y-%m-%dT%H%M%S%Z00\") -replace '[.][0-9]*_', '_'
```

Output was:

`timestamp`

In [33]:
```
#  ref1 = "https://web.archive.org/web/20240530051418/" + \
#+        "https://stackoverflow.com/questions/73221277/" + \
#+        "python-hugging-face-warning"
#  ref2 = "https://web.archive.org/web/20240530051559/" + \
#+        "https://huggingface.co/docs/transformers/en/" + \
#+        "main_classes/logging"

##  Haven't tried this, because the logging seemed easier,
##+ and the logging worked
#os.environ("TRANSFORMERS_NO_ADVISORY_WARNINGS") = 1

logging.set_verbosity_error()

summarizer = pipeline('summarization', model=model, tokenizer=tokenizer)

prediction_list = []
```

```
reference_list = []

tic = timeit.default_timer()

for sample_num in range(len(dataset['test'])):
  this_sample = dataset['test'][sample_num]

  #print(f"dialogue: \n{this_sample['dialogue']}\n---------------")

  grnd_summary = this_sample['summary']
  res = summarizer(this_sample['dialogue'])
  res_summary = res[0]['summary_text']

  #print(f"human-genratd summary:\n{grnd_summary}")
  #print(f"flan-t5-small summary:\n{res_summary}")

  reference_list.append(grnd_summary)
  prediction_list.append(res_summary)
##endof:  for sample_num in range(len(dataset['test']))

toc = timeit.default_timer()

baseline_duration = toc - tic

print( "Getting things ready for scoring")
print(f"took {toc - tic:0.4f} seconds.")

print("\n\n---------- ROUGE SCORES ----------")

rouge = load_metric('rouge', trust_remote_code=True)
  #  Set trust_remote_code=False to see the warning,
  #+ deprecation, and what to change to.

results = rouge.compute(predictions=prediction_list,
                        references=reference_list,
                        use_aggregator=True)

# >>> print(list(results.keys()))
# ['rouge1', 'rouge2', 'rougeL', 'rougeLsum']

print()
print("ROUGE-1 results")
```

```
pprint.pp(results['rouge1'])
print()
print("ROUGE-2 results")
pprint.pp(results['rouge2'])
print()
print("ROUGE-L results")
pprint.pp(results['rougeL'])
print()
print("ROUGE-Lsum results")
pprint.pp(results['rougeLsum'])

##  Haven't tried this, because the logging seemed easier,
##+ and the logging worked
# os.environ("TRANSFORMERS_NO_ADVISORY_WARNINGS") = init_t_n_a_w

logging.set_verbosity(init_log_verbosity)
```

```
Getting things ready for scoring
took 1161.2583 seconds.


---------- ROUGE SCORES ----------

ROUGE-1 results
AggregateScore(low=Score(precision=0.3629032637640585, recall=0.5391845256977906, fmeasure=0.4118077577617569), mid=S
core(precision=0.37378631220204894, recall=0.5524116283616876, fmeasure=0.4217869179887146), high=Score(precision=0.3
8500431291351583, recall=0.5650389326319216, fmeasure=0.43079470632324857))

ROUGE-2 results
AggregateScore(low=Score(precision=0.15901312161360132, recall=0.24356734238423605, fmeasure=0.18096074123365835), mi
d=Score(precision=0.16773231642486586, recall=0.256630359847149, fmeasure=0.1902131348038164), high=Score(precision=
0.1760853791540645, recall=0.27103481012172936, fmeasure=0.1993914509023913))

ROUGE-L results
AggregateScore(low=Score(precision=0.2801745713754196, recall=0.42240509773283014, fmeasure=0.31974746585335845), mid
=Score(precision=0.28942230471040165, recall=0.43518710957174245, fmeasure=0.3283896020572609), high=Score(precision=
0.2985629349792533, recall=0.44741793304388866, fmeasure=0.3369594977484363))

ROUGE-Lsum results
AggregateScore(low=Score(precision=0.2803215502826143, recall=0.42286268727590726, fmeasure=0.31922828307681683), mid
=Score(precision=0.28944780190676894, recall=0.43502677897353703, fmeasure=0.32838674112958915), high=Score(precision
=0.2989192043856499, recall=0.4482596155482941, fmeasure=0.3370955503665157))
```

In [50]:
```
#  this time, put it in manually
do_enter_duration_manually = True

if do_enter_duration_manually:
    baseline_duration = 1161.2583 # remember to type in your number, if needed
##endof:  if do_enter_duration_manually

print("Running baseline inference (using the test set)")
print(f"took {format_timespan(baseline_duration)}")
```

```
Running baseline inference (using the test set)
took 19 minutes and 21.26 seconds
```

Trainer - the Actual Trainer Part

In [37]:
```
trainer = SFTTrainer( model=model,
                      train_dataset=dataset['train'],
                      eval_dataset=dataset['evaluation'],
                      peft_config=peft_config,
                      tokenizer=tokenizer,
                      packing=True,
                      formatting_func=prompt_instruction_format,
                      args=training_args,
                    )
##  Warnings are below output.

##  Ended up not using this.
#                      max_seq_length=675
#          )
```

```
WARNING:bitsandbytes.cextension:The installed version of bitsandbytes was compiled without GPU support. 8-bit optimizers, 8-bit multiplication, and GPU quantization are unavailable.
C:\Users\Anast\.conda\envs\rwkv-lora-pat\lib\site-packages\trl\trainer\sft_trainer.py:246: UserWarning: You didn't pass a `max_seq_length` argument to the SFTTrainer, this will default to 512
  warnings.warn(
Generating train split: 0 examples [00:00, ? examples/s]
Token indices sequence length is longer than the specified maximum sequence length for this model (657 > 512). Running this sequence through the model will result in indexing errors
Generating train split: 0 examples [00:00, ? examples/s]
```

First time warnings from the code above (as it still is).

```
WARNING:bitsandbytes.cextension:The installed version of bitsandbytes \
 was compiled without GPU support. 8-bit optimizers, 8-bit multiplication, \
 and GPU quantization are unavailable.
C:\Users\bballdave025\.conda\envs\rwkv-lora-pat\lib\site-packages\trl\\
 trainer\sft_trainer.py:246: UserWarning: You didn't pass a `max_seq_length` \
argument to the SFTTrainer, this will default to 512
 warnings.warn(

[ > Generating train split: 6143/0 [00:04<00:00, 2034.36 examples/s] ]

Token indices sequence length is longer than the specified maximum sequence \
 length for this model (657 > 512). Running this sequence through the model \
 will result in indexing errors

[ > Generating train split: 355/0 [00:00<00:00, 6.10 examples/s] ]
```

DWB Note

~~So, I'm changing the `max_seq_length`~~ : Maybe I should just throw out the offender(s) (along with the blank one that's in there somewhere), but I'll just continue as is.

Actually, it appears I didn't run the updated cell, (with `max_seq_length=675`), since the Warning and Advice are still there.

Let's Train This LoRA Thing and See How It Does!

In [40]:
```
# # No need to do this again
# !powershell -c (Get-Date -UFormat \"%s_%Y-%m-%dT%H%M%S%Z00\") -replace '[.][0-9]*_', '_'
###  DWB went in and renamed `profile.ps1` to `NOT-USING_-_pro_file_-_now.ps1.bak`
###+ That should get rid of our errors from `powershell`
```

```
1717063394_2024-05-30T100314-0600
```

```
. : File C:\Users\Anast\Documents\WindowsPowerShell\profile.ps1 cannot be loaded because running scripts is disabled
on this system. For more information, see about_Execution_Policies at https:/go.microsoft.com/fwlink/?LinkID=135170.
At line:1 char:3
+ . 'C:\Users\Anast\Documents\WindowsPowerShell\profile.ps1'
+   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : SecurityError: (:) [], PSSecurityException
    + FullyQualifiedErrorId : UnauthorizedAccess
```

Output was:

`timestamp`

At about `1717063394_2024-05-30T100314-0600`, DWB went in and renamed `profile.ps1` to `NOT-USING_-_pro_file_-_now.ps1.bak` That should get rid of our errors from `powershell`

The long-time-taking training code is just below.

In [41]:
```
tic = timeit.default_timer()
trainer.train()
toc = timeit.default_timer()
print(f"tic: {tic}")
print(f"toc: {toc}")
training_duration = toc - tic
print(f"Training took {toc - tic:0.4f} seconds.")
```

[1536/1536 3:04:40, Epoch 1/1]

Step	Training Loss
500	0.302900
1000	0.077800
1500	0.066900

```
C:\Users\Anast\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\file_download.py:1132: FutureWarning: `res
ume_download` is deprecated and will be removed in version 1.0.0. Downloads always resume when possible. If you want
to force a new download, use `force_download=True`.
  warnings.warn(
```

```
tic: 329662.9144114
toc: 340750.6596588
Training took 11087.7452 seconds.
```

In [51]:
```
# this time, put it in by hand
do_by_hand = True

if do_by_hand:
  training_duration = 11087.7452 ## make sure to enter your value
##endof:  if do_by_hand
print( "Training with LoRA (and with the other info as above)")
print(f"took {format_timespan(training_duration)}.")
```

```
Training with LoRA (and with the other info as above)
took 3 hours, 4 minutes and 47.75 seconds.
```

@todo : consolidate "the other info as above"

I'm talking about the numbers of data points, tokens, whatever.

Any Comments / Things to Try (?)

We passed an evaluation set (parameter ``) to the `trainer`. How can we see information about that?

In [ ]:
```
# space for doing trainer stuff
```

In [ ]:
```
# more space for doing trainer stuff, if that happens
```

Save the Trainer to Hugging Face and Get Our Updated Model

In [52]:
```
# #Don't need this again
# !powershell -c (Get-Date -UFormat \"%s_%Y-%m-%dT%H%M%S%Z00\") -replace '[.][0-9]*_', '_'
```

```
1717078324_2024-05-30T141204-0600
```

Output was:

`1717078324_2024-05-30T141204-0600`

I'm following the (archived) tutorial from Mehul Gupta on Medium; since it's archived, you can follow exactly what I'm doing.

In [ ]:

In [59]:
```
# This will come up with a dialog box with text entry.

# Use the write token, here.
notebook_login()
```

```
VBox(children=(HTML(value='<center> <img\nsrc=https://huggingface.co/front/assets/huggingface_logo-noborder.sv…
```

In [65]:
```
# Save tokenizer and create a tokenizer model card
tokenizer.save_pretrained('testing')
  # used 'testing' first - I don't like how vague that is.
  # using 'dwb-first-lora-test-flan-t5-guptal' broke it
  #  actually, I think deleting output from my main model
  #+ page broke it.

# Create the trainer model card
trainer.create_model_card()

# Push the results to the Hugging Face Hub
trainer.push_to_hub()
```

```
C:\Users\Anast\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\file_download.py:1132: FutureWarning: `res
ume_download` is deprecated and will be removed in version 1.0.0. Downloads always resume when possible. If you want
to force a new download, use `force_download=True`.
  warnings.warn(
```

```
---------------------------------------------------------------------------
HTTPError                                 Traceback (most recent call last)
File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\utils\_errors.py:304, in hf_raise_for_status(respo
nse, endpoint_name)
    303 try:
--> 304     response.raise_for_status()
    305 except HTTPError as e:

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\requests\models.py:1024, in Response.raise_for_status(self)
   1023 if http_error_msg:
-> 1024     raise HTTPError(http_error_msg, response=self)

HTTPError: 404 Client Error: Not Found for url: https://huggingface.co/api/models/bballdave025/output/preupload/main

The above exception was the direct cause of the following exception:

RepositoryNotFoundError                   Traceback (most recent call last)
Cell In[65], line 12
      9 trainer.create_model_card()
     11 # Push the results to the Hugging Face Hub
---> 12 trainer.push_to_hub()

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\trl\trainer\sft_trainer.py:385, in SFTTrainer.push_to_hub(self, co
mmit_message, blocking, **kwargs)
    379 """
    380 Overwrite the `push_to_hub` method in order to force-add the tag "sft" when pushing the
    381 model on the Hub. Please refer to `~transformers.Trainer.push_to_hub` for more details.
    382 """
    383 kwargs = trl_sanitze_kwargs_for_tagging(model=self.model, tag_names=self._tag_names, kwargs=kwargs)
--> 385 return super().push_to_hub(commit_message=commit_message, blocking=blocking, **kwargs)

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\transformers\trainer.py:4236, in Trainer.push_to_hub(self, commit_
message, blocking, token, **kwargs)
   4234 # Wait for the current upload to be finished.
   4235 self._finish_current_push()
-> 4236 return upload_folder(
   4237     repo_id=self.hub_model_id,
   4238     folder_path=self.args.output_dir,
   4239     commit_message=commit_message,
   4240     token=token,
   4241     run_as_future=not blocking,
   4242     ignore_patterns=["_*", f"{PREFIX_CHECKPOINT_DIR}-*"],
```

```
   4243 )

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\utils\_validators.py:114, in validate_hf_hub_args.
<locals>._inner_fn(*args, **kwargs)
    111 if check_use_auth_token:
    112     kwargs = smoothly_deprecate_use_auth_token(fn_name=fn.__name__, has_token=has_token, kwargs=kwargs)
--> 114 return fn(*args, **kwargs)

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\hf_api.py:1286, in future_compatible.<locals>._inn
er(self, *args, **kwargs)
   1283     return self.run_as_future(fn, self, *args, **kwargs)
   1285 # Otherwise, call the function normally
-> 1286 return fn(self, *args, **kwargs)

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\hf_api.py:4724, in HfApi.upload_folder(self, repo_
id, folder_path, path_in_repo, commit_message, commit_description, token, repo_type, revision, create_pr, parent_comm
it, allow_patterns, ignore_patterns, delete_patterns, multi_commits, multi_commits_verbose, run_as_future)
   4720     # Defining a CommitInfo object is not really relevant in this case
   4721     # Let's return early with pr_url only (as string).
   4722     return pr_url
-> 4724 commit_info = self.create_commit(
   4725     repo_type=repo_type,
   4726     repo_id=repo_id,
   4727     operations=commit_operations,
   4728     commit_message=commit_message,
   4729     commit_description=commit_description,
   4730     token=token,
   4731     revision=revision,
   4732     create_pr=create_pr,
   4733     parent_commit=parent_commit,
   4734 )
   4736 # Create url to uploaded folder (for legacy return value)
   4737 if create_pr and commit_info.pr_url is not None:

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\utils\_validators.py:114, in validate_hf_hub_args.
<locals>._inner_fn(*args, **kwargs)
    111 if check_use_auth_token:
    112     kwargs = smoothly_deprecate_use_auth_token(fn_name=fn.__name__, has_token=has_token, kwargs=kwargs)
--> 114 return fn(*args, **kwargs)

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\hf_api.py:1286, in future_compatible.<locals>._inn
er(self, *args, **kwargs)
```

```
   1283     return self.run_as_future(fn, self, *args, **kwargs)
   1285 # Otherwise, call the function normally
-> 1286 return fn(self, *args, **kwargs)

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\hf_api.py:3677, in HfApi.create_commit(self, repo_id, operations, commit_message, commit_description, token, repo_type, revision, create_pr, num_threads, parent_commit, run_as_future)
   3674 # If updating twice the same file or update then delete a file in a single commit
   3675 _warn_on_overwriting_operations(operations)
-> 3677 self.preupload_lfs_files(
   3678     repo_id=repo_id,
   3679     additions=additions,
   3680     token=token,
   3681     repo_type=repo_type,
   3682     revision=unquoted_revision,  # first-class methods take unquoted revision
   3683     create_pr=create_pr,
   3684     num_threads=num_threads,
   3685     free_memory=False,  # do not remove `CommitOperationAdd.path_or_fileobj` on LFS files for "normal" users
   3686 )
   3687 files_to_copy = _fetch_files_to_copy(
   3688     copies=copies,
   3689     repo_type=repo_type,
   (...)
   3693     endpoint=self.endpoint,
   3694 )
   3695 commit_payload = _prepare_commit_payload(
   3696     operations=operations,
   3697     files_to_copy=files_to_copy,
   (...)
   3700     parent_commit=parent_commit,
   3701 )

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\hf_api.py:4153, in HfApi.preupload_lfs_files(self, repo_id, additions, token, repo_type, revision, create_pr, num_threads, free_memory, gitignore_content)
   4151 # Check which new files are LFS
   4152 try:
-> 4153     _fetch_upload_modes(
   4154         additions=new_additions,
   4155         repo_type=repo_type,
   4156         repo_id=repo_id,
   4157         headers=headers,
   4158         revision=revision,
```

```
   4159         endpoint=self.endpoint,
   4160         create_pr=create_pr or False,
   4161         gitignore_content=gitignore_content,
   4162     )
   4163 except RepositoryNotFoundError as e:
   4164     e.append_to_message(_CREATE_COMMIT_NO_REPO_ERROR_MESSAGE)

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\utils\_validators.py:114, in validate_hf_hub_args.
<locals>._inner_fn(*args, **kwargs)
    111 if check_use_auth_token:
    112     kwargs = smoothly_deprecate_use_auth_token(fn_name=fn.__name__, has_token=has_token, kwargs=kwargs)
--> 114 return fn(*args, **kwargs)

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\_commit_api.py:508, in _fetch_upload_modes(additio
ns, repo_type, repo_id, headers, revision, endpoint, create_pr, gitignore_content)
    500     payload["gitIgnore"] = gitignore_content
    502 resp = get_session().post(
    503     f"{endpoint}/api/{repo_type}s/{repo_id}/preupload/{revision}",
    504     json=payload,
    505     headers=headers,
    506     params={"create_pr": "1"} if create_pr else None,
    507 )
--> 508 hf_raise_for_status(resp)
    509 preupload_info = _validate_preupload_info(resp.json())
    510 upload_modes.update(**{file["path"]: file["uploadMode"] for file in preupload_info["files"]})

File ~\.conda\envs\rwkv-lora-pat\lib\site-packages\huggingface_hub\utils\_errors.py:352, in hf_raise_for_status(respo
nse, endpoint_name)
    333 elif error_code == "RepoNotFound" or (
    334     response.status_code == 401
    335     and response.request is not None
   (...)
    342     # => for now, we process them as `RepoNotFound` anyway.
    343     # See https://gist.github.com/Wauplin/46c27ad266b15998ce56a6603796f0b9
    344     message = (
    345         f"{response.status_code} Client Error."
    346         + "\n\n"
   (...)
    350         " make sure you are authenticated."
    351     )
--> 352     raise RepositoryNotFoundError(message, response) from e
    354 elif response.status_code == 400:
```

```
    355     message = (
    356         f"\n\nBad request for {endpoint_name} endpoint:" if endpoint_name is not None else "\n\nBad request:"
    357     )

RepositoryNotFoundError: 404 Client Error. (Request ID: Root=1-6658ed74-3a7f95084ece914c5a48a925;36d7e7a9-7c0b-442c-9
350-0cfb774bff0c)

Repository Not Found for url: https://huggingface.co/api/models/bballdave025/output/preupload/main.
Please make sure you specified the correct `repo_id` and `repo_type`.
If you are trying to access a private or gated repo, make sure you are authenticated.
Note: Creating a commit assumes that the repo already exists on the Huggingface Hub. Please use `create_repo` if it's
not the case.
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Evaluation on the Test Set and Comparison to Baseline

Verbosity stuff - get rid of the nice advice

In [ ]:
```
!powershell -c (Get-Date -UFormat \"%s_%Y-%m-%dT%H%M%S%Z00\") -replace '[.][0-9]*_', '_'
```

Output was:

```
timestamp
```

In [ ]:
```
# bballdave025@MYMACHINE /cygdrive/c/Users/bballdave025/.conda/envs/rwkv-lora-pat/Lib/site-packages/peft/utils
# $ date +'%s_%Y-%m-%dT%H%M%S%z'
```

```
# 1717049876_2024-05-30T001756-0600

log_verbosity_is_critical = \
  logging.get_verbosity() == logging.CRITICAL # alias FATAL, 50
log_verbosity_is_error = \
  logging.get_verbosity() == logging.ERROR # 40
log_verbosity_is_warn = \
  logging.get_verbosity() == logging.WARNING # alias WARN, 30
log_verbosity_is_info = \
  logging.get_verbosity() == logging.INFO # 20
log_verbosity_is_debug = \
  logging.get_verbosity() == logging.DEBUG # 10

print( "The statement, 'logging verbosity is CRITICAL' " + \
      f"is {log_verbosity_is_critical}")
print( "The statement, 'logging verbosity is    ERROR' " + \
      f"is {log_verbosity_is_error}")
print( "The statement, 'logging verbosity is  WARNING' " + \
      f"is {log_verbosity_is_warn}")
print( "The statement, 'logging verbosity is     INFO' " + \
      f"is {log_verbosity_is_info}")
print( "The statement, 'logging verbosity is    DEBUG' " + \
      f"is {log_verbosity_is_debug}")

print()

init_log_verbosity = logging.get_verbosity()
print(f"The value of logging.get_verbosity() is: {init_log_verbosity}")

print()

init_t_n_a_w = os.environ.get('TRANSFORMERS_NO_ADVISORY_WARNINGS')
print(f"TRANSFORMERS_NO_ADIVSORY_WARNINGS: {init_t_n_a_w}")
```

Here's the actual evaluation

In [ ]:
```
!powershell -c (Get-Date -UFormat \"%s_%Y-%m-%dT%H%M%S%Z00\") -replace '[.][0-9]*_', '_'
```

Output was:

`timestamp`

!!! NOTE !!! I'm going to use `tat` (with an underscore or undescores before, after, or surrounding the variable names) to indicate 'testing-after-training'.

In [ ]:
```
#  I'm going to use 'tat' for testing-after-training

logging.set_verbosity_error()

summarizer = pipeline('summarization', model=model, tokenizer=tokenizer)

prediction_tat_list = []
reference_tat_list = []

tic = timeit.default_timer()

for sample_num in range(len(dataset['test'])):
  this_sample = dataset['test'][sample_num]

  #print(f"dialogue: \n{this_sample['dialogue']}\n---------------")

  grnd_tat_summary = this_sample['summary']
  res_tat = summarizer(this_sample['dialogue'])
  res_tat_summary = res_tat[0]['summary_text']

  #print(f"human-genratd summary:\n{grnd_tat_summary}")
  #print(f"flan-t5-small summary:\n{res_tat_summary}")

  reference_tat_list.append(grnd_tat_summary)
  prediction_tat_list.append(res_tat_summary)
##endof:  for sample_num in range(len(dataset['test']))

toc = timeit.default_timer()

print( "Getting things ready for scoring (after training)")
print(f"took {toc - tic:0.4f} seconds.")

print("\n\n---------- ROUGE SCORES ----------")

rouge = load_metric('rouge', trust_remote_code=True)
  #  Set trust_remote_code=False to see the warning,
```

```
  #+ deprecation, and what to change to.

results_tat = rouge.compute(
                  predictions=prediction_tat_list,
                  references=reference_tat_list,
                  use_aggregator=True
)

# >>> print(list(results_tat.keys()))
# ['rouge1', 'rouge2', 'rougeL', 'rougeLsum']

print()
print("ROUGE-1 results")
pprint.pp(results_tat['rouge1'])
print()
print("ROUGE-2 results")
pprint.pp(results_tat['rouge2'])
print()
print("ROUGE-L results")
pprint.pp(results_tat['rougeL'])
print()
print("ROUGE-Lsum results")
pprint.pp(results_tat['rougeLsum'])

logging.set_verbosity(init_log_verbosity)
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: