In [22]:
```
!pip install sentencepiece
!pip install transformers
!pip install rich[jupyter]
```

```
Looking in indexes: https://pypi.org/simple, (https://pypi.org/simple,) http
s://us-python.pkg.dev/colab-wheels/public/simple/ (https://us-python.pkg.dev/
colab-wheels/public/simple/)
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.10/dis
t-packages (0.1.99)
Looking in indexes: https://pypi.org/simple, (https://pypi.org/simple,) http
s://us-python.pkg.dev/colab-wheels/public/simple/ (https://us-python.pkg.dev/
colab-wheels/public/simple/)
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist
-packages (4.29.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-pac
kages (from transformers) (3.12.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.11.0 in /usr/local/lib
/python3.10/dist-packages (from transformers) (0.14.1)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from transformers) (1.22.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/d
ist-packages (from transformers) (23.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
```

In [23]:
```
import pandas as pd
df = pd.read_csv("/content/Conversation_Chatbot.csv")
```

In [24]:
```
df.sample(10)
```

Out[24]:

	0	question	answer
83	84	What is the "Whiffenpoofs" at Yale University?	The "Whiffenpoofs" is the oldest collegiate a ...
84	85	The "Whiffenpoofs" is the oldest collegiate a ...	It's so nice
42	43	Yes, It's an interdisciplinary learning	Can, you elaborate more about the course?
108	109	Can you tell me for any grad program point of ...	It generally requires submitting transcripts, ...
20	21	it's not bad. there are a lot of people there.	good luck with that.
24	25	i'm absolutely lovely, thank you.	everything's been good with you?
5	6	i've been good. i'm in school right now.	what school do you go to?
1	2	i'm fine. how about yourself?	i'm pretty good. thanks for asking.
74	75	That's seems to be very tough	Yes, it is a highly reputed college.
126	127	It varies depending on the program.	Can you tell me for any grad program point of ...

In [25]:
```
df["answer"] = "summarize: "+df["answer"]
```

In [26]:
```
df.head()
```

Out[26]:

	0	question	answer
0	1	hi, how are you doing?	summarize: i'm fine. how about yourself?
1	2	i'm fine. how about yourself?	summarize: i'm pretty good. thanks for asking.
2	3	i'm pretty good. thanks for asking.	summarize: no problem. so how have you been?
3	4	no problem. so how have you been?	summarize: i've been great. what about you?
4	5	i've been great. what about you?	summarize: i've been good. i'm in school right...

In [27]:
```
# Importing libraries
import os
import numpy as np
import pandas as pd
import torch
import torch.nn.functional as F
from torch.utils.data import Dataset, DataLoader, RandomSampler, SequentialSam
import os

# Importing the T5 modules from huggingface/transformers
from transformers import T5Tokenizer, T5ForConditionalGeneration

from rich.table import Column, Table
from rich import box
from rich.console import Console

# define a rich console logger
console=Console(record=True)

def display_df(df):
  """display dataframe in ASCII format"""

  console=Console()
  table = Table(Column("source_text", justify="center" ), Column("target_text"

  for i, row in enumerate(df.values.tolist()):
    table.add_row(row[0], row[1])

  console.print(table)

training_logger = Table(Column("Epoch", justify="center" ),
                        Column("Steps", justify="center"),
                        Column("Loss", justify="center"),
                        title="Training Status",pad_edge=False, box=box.ASCII)
```

In [28]:
```
# Setting up the device for GPU usage
from torch import cuda
device = 'cuda' if cuda.is_available() else 'cpu'
```

In [29]:
```
class YourDataSetClass(Dataset):
  """
  Creating a custom dataset for reading the dataset and
  loading it into the dataloader to pass it to the neural network for finetuni

  """

  def __init__(self, dataframe, tokenizer, source_len, target_len, source_text
    self.tokenizer = tokenizer
    self.data = dataframe
    self.source_len = source_len
    self.summ_len = target_len
    self.target_text = self.data[target_text]
    self.source_text = self.data[source_text]

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

  def __getitem__(self, index):
    source_text = str(self.source_text[index])
    target_text = str(self.target_text[index])

    #cleaning data so as to ensure data is in string type
    source_text = ' '.join(source_text.split())
    target_text = ' '.join(target_text.split())

    source = self.tokenizer.batch_encode_plus([source_text], max_length= self.
    target = self.tokenizer.batch_encode_plus([target_text], max_length= self.

    source_ids = source['input_ids'].squeeze()
    source_mask = source['attention_mask'].squeeze()
    target_ids = target['input_ids'].squeeze()
    target_mask = target['attention_mask'].squeeze()

    return {
        'source_ids': source_ids.to(dtype=torch.long),
        'source_mask': source_mask.to(dtype=torch.long),
        'target_ids': target_ids.to(dtype=torch.long),
        'target_ids_y': target_ids.to(dtype=torch.long)
    }
```

In [30]:
```
def train(epoch, tokenizer, model, device, loader, optimizer):

  """
  Function to be called for training with the parameters passed from main func

  """

  model.train()
  for _,data in enumerate(loader, 0):
    y = data['target_ids'].to(device, dtype = torch.long)
    y_ids = y[:, :-1].contiguous()
    lm_labels = y[:, 1:].clone().detach()
    lm_labels[y[:, 1:] == tokenizer.pad_token_id] = -100
    ids = data['source_ids'].to(device, dtype = torch.long)
    mask = data['source_mask'].to(device, dtype = torch.long)

    outputs = model(input_ids = ids, attention_mask = mask, decoder_input_ids=
    loss = outputs[0]

    if _%10==0:
      training_logger.add_row(str(epoch), str(_), str(loss))
      console.print(training_logger)

    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
```

In [31]:
```
def validate(epoch, tokenizer, model, device, loader):

  """
  Function to evaluate model for predictions

  """
  model.eval()
  predictions = []
  actuals = []
  with torch.no_grad():
      for _, data in enumerate(loader, 0):
          y = data['target_ids'].to(device, dtype = torch.long)
          ids = data['source_ids'].to(device, dtype = torch.long)
          mask = data['source_mask'].to(device, dtype = torch.long)

          generated_ids = model.generate(
              input_ids = ids,
              attention_mask = mask,
              max_length=150,
              num_beams=2,
              repetition_penalty=2.5,
              length_penalty=1.0,
              early_stopping=True
              )
          preds = [tokenizer.decode(g, skip_special_tokens=True, clean_up_toke
          target = [tokenizer.decode(t, skip_special_tokens=True, clean_up_tok
          if _%10==0:
              console.print(f'Completed {_}')

          predictions.extend(preds)
          actuals.extend(target)
  return predictions, actuals
```

In [31]:

In [32]:
```
def T5Trainer(dataframe, source_text, target_text, model_params, output_dir=".

    """
    T5 trainer

    """

    # Set random seeds and deterministic pytorch for reproducibility
    torch.manual_seed(model_params["SEED"]) # pytorch random seed
    np.random.seed(model_params["SEED"]) # numpy random seed
    torch.backends.cudnn.deterministic = True

    # logging
    console.log(f"""[Model]: Loading {model_params["MODEL"]}...\n""")

    # tokenzier for encoding the text
    tokenizer = T5Tokenizer.from_pretrained(model_params["MODEL"])

    # Defining the model. We are using t5-base model and added a Language model
    # Further this model is sent to device (GPU/TPU) for using the hardware.
    model = T5ForConditionalGeneration.from_pretrained(model_params["MODEL"])
    model = model.to(device)

    # logging
    console.log(f"[Data]: Reading data...\n")

    # Importing the raw dataset
    dataframe = dataframe[[source_text,target_text]]
    display_df(dataframe.head(2))


    # Creation of Dataset and Dataloader
    # Defining the train size. So 80% of the data will be used for training and
    train_size = 0.8
    train_dataset=dataframe.sample(frac=train_size,random_state = model_params["
    val_dataset=dataframe.drop(train_dataset.index).reset_index(drop=True)
    train_dataset = train_dataset.reset_index(drop=True)

    console.print(f"FULL Dataset: {dataframe.shape}")
    console.print(f"TRAIN Dataset: {train_dataset.shape}")
    console.print(f"TEST Dataset: {val_dataset.shape}\n")


    # Creating the Training and Validation dataset for further creation of Datal
    training_set = YourDataSetClass(train_dataset, tokenizer, model_params["MAX_
    val_set = YourDataSetClass(val_dataset, tokenizer, model_params["MAX_SOURCE_


    # Defining the parameters for creation of dataloaders
    train_params = {
        'batch_size': model_params["TRAIN_BATCH_SIZE"],
        'shuffle': True,
        'num_workers': 0
        }
    val_params = {
        'batch_size': model_params["VALID_BATCH_SIZE"],
        'shuffle': False,
        'num_workers': 0
        }


    # Creation of Dataloaders for testing and validation. This will be used down
    training_loader = DataLoader(training_set, **train_params)
    val_loader = DataLoader(val_set, **val_params)


    # Defining the optimizer that will be used to tune the weights of the networ
    optimizer = torch.optim.Adam(params =  model.parameters(), lr=model_params["


    # Training loop
    console.log(f'[Initiating Fine Tuning]...\n')

    for epoch in range(model_params["TRAIN_EPOCHS"]):
        train(epoch, tokenizer, model, device, training_loader, optimizer)

    console.log(f"[Saving Model]...\n")
    #Saving the model after training
    path = os.path.join(output_dir, "model_files")
    model.save_pretrained(path)
    tokenizer.save_pretrained(path)


    # evaluating test dataset
    console.log(f"[Initiating Validation]...\n")
    for epoch in range(model_params["VAL_EPOCHS"]):
      predictions, actuals = validate(epoch, tokenizer, model, device, val_loade
      final_df = pd.DataFrame({'Generated Text':predictions,'Actual Text':actual
      final_df.to_csv(os.path.join(output_dir,'predictions.csv'))

    console.save_text(os.path.join(output_dir,'logs.txt'))

    console.log(f"[Validation Completed.]\n")
    console.print(f"""[Model] Model saved @ {os.path.join(output_dir, "model_fil
    console.print(f"""[Validation] Generation on Validation data saved @ {os.pat
    console.print(f"""[Logs] Logs saved @ {os.path.join(output_dir,'logs.txt')}\
```

In [33]:
```
model_params={
    "MODEL":"t5-base",             # model_type: t5-base/t5-large
    "TRAIN_BATCH_SIZE":8,          # training batch size
    "VALID_BATCH_SIZE":8,          # validation batch size
    "TRAIN_EPOCHS":3,              # number of training epochs
    "VAL_EPOCHS":1,                # number of validation epochs
    "LEARNING_RATE":1e-4,          # learning rate
    "MAX_SOURCE_TEXT_LENGTH":512,  # max length of source text
    "MAX_TARGET_TEXT_LENGTH":50,   # max length of target text
    "SEED": 42                   # set seed for reproducibility

}
```

```
In [34]: T5Trainer(dataframe=df[:127], source_text="question", target_text="answer", mo
```

```
[02:49:44] [Model]: Loading t5-base...


/usr/local/lib/python3.10/dist-packages/transformers/models/t5/tokenization_t
5.py:163: FutureWarning: This tokenizer was incorrectly instantiated with a m
odel max length of 512 which will be corrected in Transformers v5.
For now, this behavior is kept to avoid breaking backwards compatibility when
padding/encoding with `truncation is True`.
- Be aware that you SHOULD NOT rely on t5-base automatically truncating your
input to 512 when padding/encoding.
- If you want to encode/pad to sequences longer than 512 you can either insta
ntiate this tokenizer with `model_max_length` or pass `max_length` when encod
ing/padding.
- To avoid this warning, please instantiate this tokenizer with `model_max_le
ngth` set to your preferred value.
  warnings.warn(

[02:49:48] [Data]: Reading data...


                                 Sample Data
+---------------------------------------------------------------------------
|        source_text            |                target_text
|-------------------------------+-------------------------------------------
|   hi, how are you doing?      |    summarize: i'm fine. how about yourself?
|i'm fine. how about yourself?  | summarize: i'm pretty good. thanks for askin
+---------------------------------------------------------------------------

FULL Dataset: (127, 2)

TRAIN Dataset: (102, 2)

TEST Dataset: (25, 2)


           [Initiating Fine Tuning]...


                              Training Status
+----------------------------------------------------------------------------+
|Epoch | Steps |                         Loss                                |
|------+-------+-------------------------------------------------------------|
|  0   |   0   | tensor(6.8346, device='cuda:0', grad_fn=<NllLossBackward0>)|
+----------------------------------------------------------------------------+

                              Training Status
+----------------------------------------------------------------------------+
|Epoch | Steps |                         Loss                                |
|------+-------+-------------------------------------------------------------|
|  0   |   0   | tensor(6.8346, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  0   |  10   | tensor(4.1578, device='cuda:0', grad_fn=<NllLossBackward0>)|
+----------------------------------------------------------------------------+

                              Training Status
+----------------------------------------------------------------------------+
|Epoch | Steps |                         Loss                                |
|------+-------+-------------------------------------------------------------|
|  0   |   0   | tensor(6.8346, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  0   |  10   | tensor(4.1578, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  1   |   0   | tensor(4.4832, device='cuda:0', grad_fn=<NllLossBackward0>)|
+------------------------------------------------------------------------------+

                              Training Status
+------------------------------------------------------------------------------+
|Epoch | Steps |                            Loss                              |
|------+-------+--------------------------------------------------------------|
|  0   |   0   | tensor(6.8346, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  0   |  10   | tensor(4.1578, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  1   |   0   | tensor(4.4832, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  1   |  10   | tensor(3.5538, device='cuda:0', grad_fn=<NllLossBackward0>)|
+------------------------------------------------------------------------------+

                              Training Status
+------------------------------------------------------------------------------+
|Epoch | Steps |                            Loss                              |
|------+-------+--------------------------------------------------------------|
|  0   |   0   | tensor(6.8346, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  0   |  10   | tensor(4.1578, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  1   |   0   | tensor(4.4832, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  1   |  10   | tensor(3.5538, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  2   |   0   | tensor(3.2713, device='cuda:0', grad_fn=<NllLossBackward0>)|
+------------------------------------------------------------------------------+

                              Training Status
+------------------------------------------------------------------------------+
|Epoch | Steps |                            Loss                              |
|------+-------+--------------------------------------------------------------|
|  0   |   0   | tensor(6.8346, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  0   |  10   | tensor(4.1578, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  1   |   0   | tensor(4.4832, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  1   |  10   | tensor(3.5538, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  2   |   0   | tensor(3.2713, device='cuda:0', grad_fn=<NllLossBackward0>)|
|  2   |  10   | tensor(2.7085, device='cuda:0', grad_fn=<NllLossBackward0>)|
+------------------------------------------------------------------------------+

[02:50:25] [Saving Model]...


[02:50:28] [Initiating Validation]...


Completed 0

[02:50:32] [Validation Completed.]


[Model] Model saved @ outputs/model_files


[Validation] Generation on Validation data saved @ outputs/predictions.csv


[Logs] Logs saved @ outputs/logs.txt
```

```
In [34]:
```

```
In [34]:
```