ECS7022P: Computational Creativity Project

Project Title: **Star Trek Dialogue Generator**
Student Name: **Rohan Ajay Deuskar**
Student ID Number: **ec22054** (220217275)

HTML Link to Colab Notebook [required]: Star Trek GPT model

Project Overview [10%]

Having been a star trek fan since young age, I have always wanted more from the world of USS Enterprise, and now that we have so many movies and shows themed around the star trek universe, with the power of A.I. we can have an infinite number of stories come from the universe. That has been my motivation for this project.

I started out by scouting to find source files for the movie scripts and landed on https://www.st-minutiae.com/ which had an archive of scripts from published sources on some Star Trek shows and movies. I decided on using the Star Trek: The Next Generation scripts.

For selecting a model I started out with a simple RNN model, more specifically a LSTM model, however after a lot of training, the outputs I got were not satisfactory. I also tried using a BERT model, with no avail. After doing a little bit of research I found that a GPT model has much been able to achieve much better results than traditional RNNs, hence I decided to try the GPT-2 model which was pretrained and available on Hugging Face. After testing different techniques and methods to train the model (within the confines of Google Colab's GPU resources) I was able to use https://huggingface.co/docs/transformers/v4.28.1/en/training as my primary resource to understand how to fine-tune the model.

I fine-tuned the model on dialogues from the show which were extracted from the scripts and then pre-processed.

Finally using a python Gradio interface I built an UI which one can use to generate snippets of conversations from the show.

Generative Models [10%]

For the purpose of the project, I employed a GPT-2 model https://github.com/openai/gpt-2 model which is available through the hugging-face transformers module in python.
Using this model and the gpt-2 tokenizer that the module provides I created a fine-tuned model which was trained on the star-trek dialogues.

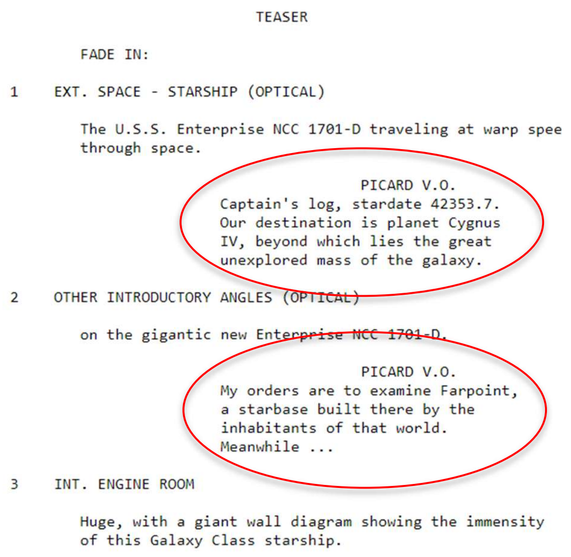
For the data used to train the model I wrote some regular expression and string manipulation code to extract the dialogues from the scripts of all episodes found in the data. This data was then split into training and testing(evaluation) using sklearn's split function with a ratio of 3:17. After pre-processing and tokenizing, this data was used to set up the Trainer class which the transformer module provides to train the models. Some trial and error were needed to change the batch-size of the inputs and the block size of each data object to get it to run using the amount of GPU Colab provides.

The dialogues provided were arranged as `character: dialogue`, Hence after fine-tuning the model the outputs it could produce were also in that format. This made it easy to set up an interface to get a character a starting dialogue as input and then let the model predict the rest.

The outputs of the model weren't quite what I expected, reason lies on the fact that I cannot train bigger chunks of the scripts and let the model understand the coherence between the dialogues, hence most of the outputs were very random conversations, however, the model was able to identify niche dialogues like Picard's voice overs saying, 'Captain's log star-date …' and such. There is potential in the system however given the constraints it does best what it can achieve.

Process [15%]

For the project, I extracted dialogues from the scripts using regular expressions to find the places where the dialogue starts and ends.



```
                              TEASER

          FADE IN:

1    EXT. SPACE - STARSHIP (OPTICAL)

          The U.S.S. Enterprise NCC 1701-D traveling at warp spee
          through space.

                                    PICARD V.O.
                         Captain's log, stardate 42353.7.
                         Our destination is planet Cygnus
                         IV, beyond which lies the great
                         unexplored mass of the galaxy.

2    OTHER INTRODUCTORY ANGLES (OPTICAL)

          on the gigantic new Enterprise NCC 1701-D.

                                    PICARD V.O.
                         My orders are to examine Farpoint,
                         a starbase built there by the
                         inhabitants of that world.
                         Meanwhile ...

3    INT. ENGINE ROOM

          Huge, with a giant wall diagram showing the immensity
          of this Galaxy Class starship.
```

After extracting this data, I pre-processed it in the form of `character: dialogue` to feed the model during the training. I split the data into two parts, one for training and one for evaluation and converted them to Dataset objects and created a Data Collator object which takes care of padding the text to block size. The hugging face documentation provides good examples and was used for the purpose of this task.

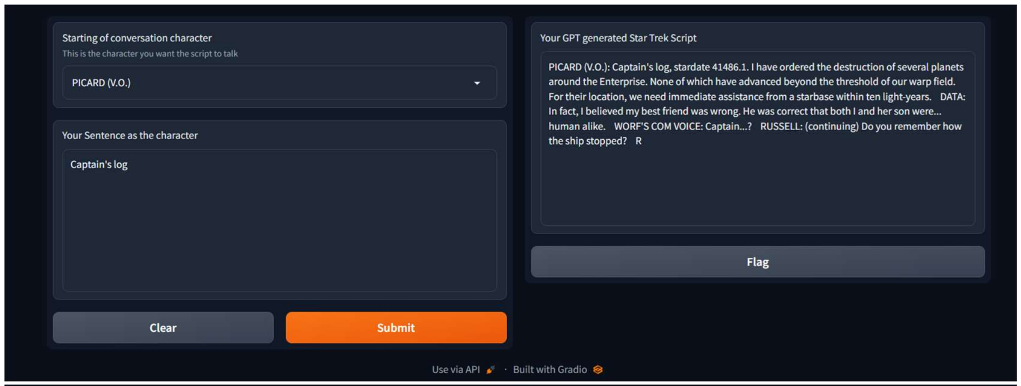
Once the data was prepared, I created a model from the pre-trained GPT-2 model, which was then passed to the Trainer object along with a TrainerArguments object, the test dataset, the evaluation dataset, and the data collator. For the optimiser, I used the adamw_torch optimiser which PyTorch provides. Yet again, the transformer library takes care of configuring the training loop and provides a feature complete training using PyTorch as its base. All I had to do after the setup was call the train function on the Trainer object. There was some trial and error needed for this as the initial batch size and block size resulted in out of memory errors.

When the model was trained, I saved it to my google drive and then tested it using command line. After I had found satisfactory results, I moved on to create a user interface to use the model.

Using Gradio, I built a simple interface which would let me generate the dialogue artefacts from the model. I used the text-generation pipeline provided by transformers module for this and passed it my fine-tuned model and the base tokenizer.

Example Outputs [10%]

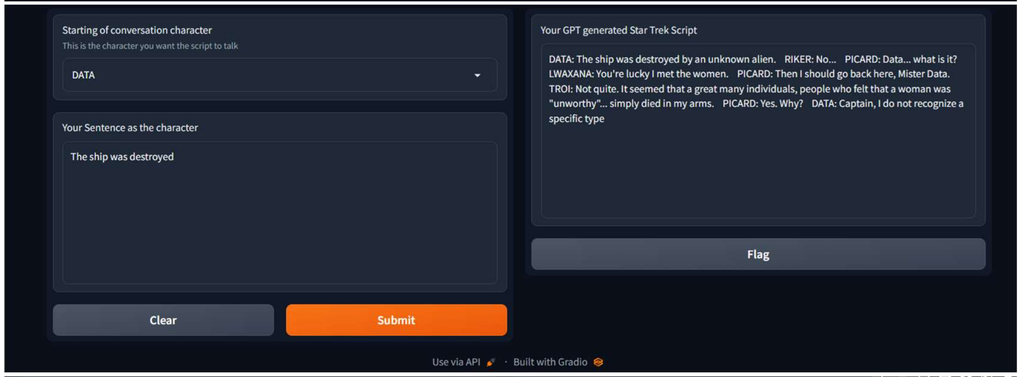
The first dropdown lets you choose a character from the scripts its trained on. The textbox below it sets some starting dialogue which would be attached to the character at the start of the script.
Finally the text box to the right calls the infer function which passes the starting dialogue by the character to the pipeline where the model is running and gets an output, which is displayed there.



Starting of conversation character
This is the character you want the script to talk
PICARD (V.O.)

Your Sentence as the character
Captain's log

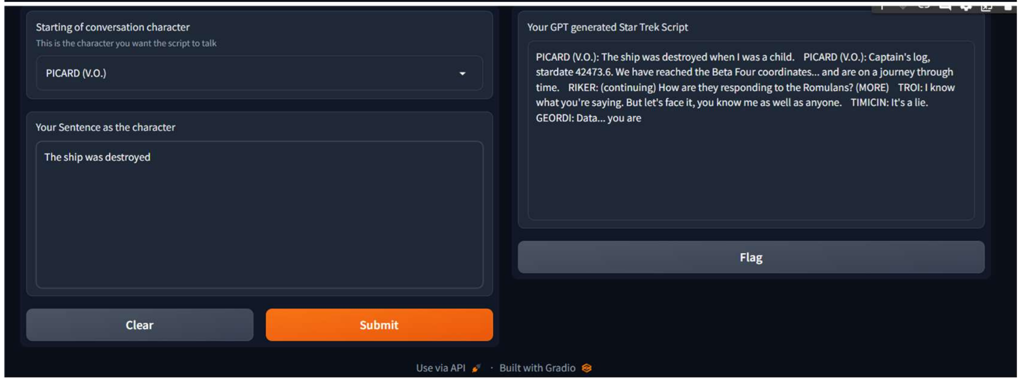
Your GPT generated Star Trek Script
PICARD (V.O.): Captain's log, stardate 41486.1. I have ordered the destruction of several planets around the Enterprise. None of which have advanced beyond the threshold of our warp field. For their location, we need immediate assistance from a starbase within ten light-years.   DATA: In fact, I believed my best friend was wrong. He was correct that both I and her son were... human alike.   WORF'S COM VOICE: Captain...?   RUSSELL: (continuing) Do you remember how the ship stopped?   R



Starting of conversation character
This is the character you want the script to talk
DATA

Your Sentence as the character
The ship was destroyed

Your GPT generated Star Trek Script
DATA: The ship was destroyed by an unknown alien.   RIKER: No...   PICARD: Data... what is it?   LWAXANA: You're lucky I met the women.   PICARD: Then I should go back here, Mister Data.   TROI: Not quite. It seemed that a great many individuals, people who felt that a woman was "unworthy"... simply died in my arms.   PICARD: Yes. Why?   DATA: Captain, I do not recognize a specific type



Starting of conversation character
This is the character you want the script to talk
PICARD (V.O.)

Your Sentence as the character
The ship was destroyed

Your GPT generated Star Trek Script
PICARD (V.O.): The ship was destroyed when I was a child.   PICARD (V.O.): Captain's log, stardate 42473.6. We have reached the Beta Four coordinates... and are on a journey through time.   RIKER: (continuing) How are they responding to the Romulans? (MORE)   TROI: I know what you're saying. But let's face it, you know me as well as anyone.   TIMICIN: It's a lie.   GEORDI: Data... you are

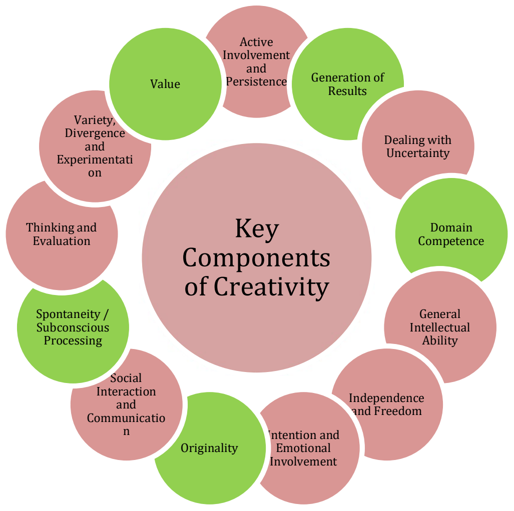
Evaluation [20%]

For the evaluation of my project, I chose to follow the Standardised Procedure for Evaluating Creative Systems (SPECS) framework.

1. **Identify a definition of creativity that your system should satisfy to be considered creative?**
   For my system I consider the Boden's Exploratory Creativity concept. A creative system which comes up with new idea in an existing conceptual space. A good example of this for reference to my project is the Nothing, Forever project.

My project is an exploratory system in the domain of the Star Trek universe. It expands the existing space of the universe by providing never-ending conversations from it.

2. **Using Step 1, clearly state what standards you use to evaluate the creativity of your system?**
I decided to use 5 of the 14 components that SPECS proposes as my standards as they fit the concept of the definition.


Key Components of Creativity
Active Involvement and Persistence
Generation of Results
Dealing with Uncertainty
Domain Competence
General Intellectual Ability
Independence and Freedom
Intention and Emotional Involvement
Originality
Social Interaction and Communication
Spontaneity / Subconscious Processing
Thinking and Evaluation
Variety, Divergence and Experimentation
Value

3. **Test your creative system against the standards stated in Step 2 and report the results.**
Generation of Results: Is the system able to generate new and interesting Star Trek dialogues that are consistent with the characters of the TV series?
No, it can generate dialogues; however, they are not consistent with the characters.

Domain Competence: Is it able to recognize the characters, settings, and overall themes of the show, and generate dialogues that are consistent with these elements?
Yes, the artefacts are consistent with the characters and settings of the show.

Originality: Does it generate dialogues that are not simply a repetition of existing dialogues, but instead adds new and interesting elements to the universe?
Yes, every dialogue it generates is new and interesting to the universe.

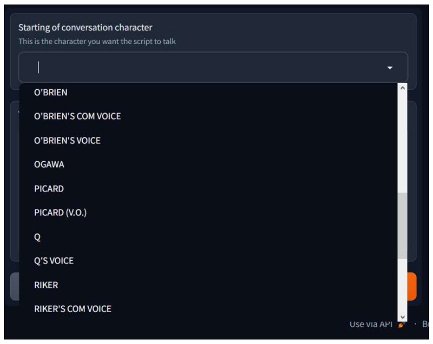
Spontaneity: Does it generate dialogues that feel natural and fluid, rather than sounding like pre-scripted responses?
Neutral, the dialogues do feel natural however they don't flow well, conversations jump on and off topic a lot.

Value: Are the dialogues engaging and entertaining, and capture the spirit of the Star Trek universe?
Yes, the dialogues are engaging and do capture the spirit of the Star Trek universe.

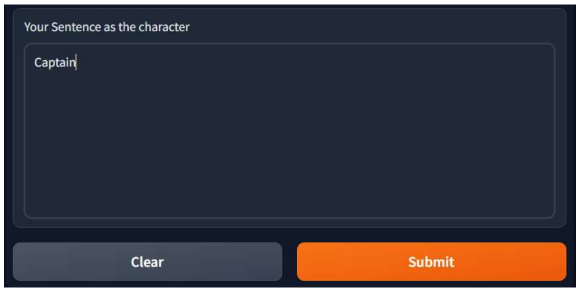
Overall, the system can be considered creative and is able to generate interesting artefacts which haven't been seen in the context of the tv show.

Value Added [20%]

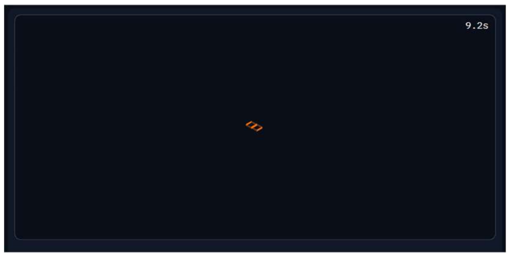
For adding value to the trained model, I tried creating a casual user-interface which lets users generate dialogues from their favourite characters from the show. The interface lets you select a character from a dropdown list.

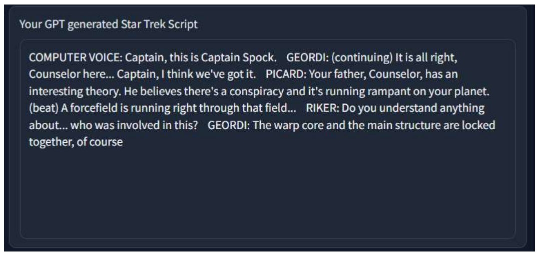


There is a textbox which lets you type some starting dialogues by the character.



Finally clicking on the submit button sends that data to the infer function which asks the text-generation pipeline to generate text via the fine-tuned GPT model and get an output.





COMPUTER VOICE: Captain, this is Captain Spock. GEORDI: (continuing) It is all right, Counselor here... Captain, I think we've got it. PICARD: Your father, Counselor, has an interesting theory. He believes there's a conspiracy and it's running rampant on your planet. (beat) A forcefield is running right through that field... RIKER: Do you understand anything about... who was involved in this? GEORDI: The warp core and the main structure are locked together, of course

Currently the model is imported via google drive, however it can be uploaded to GitHub and imported via a clone command for use by anybody. I intend to do so later in the future.

Colab Notebook Code [15%]

```
"""CC GPT StarTrek TNG.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1F_j-IDSPX3Sw8G3oOywkV4I8pmvNFMAR

#Installations for all modules
"""

!pip install transformers gradio
import json
from torch.utils.data import Dataset
from transformers import GPT2LMHeadModel, GPT2Tokenizer
from torch.optim import Adam
from torch.utils.data import DataLoader
import tqdm
import torch
import pandas as pd
import numpy as np
from torch.cuda import amp
from sklearn.model_selection import train_test_split
import gradio as gr

"""##Mounting Drive for saving model"""

from google.colab import drive
drive.mount('/content/gdrive')

"""#Reading and pre-processing scripts to fine-tune GPT-2 model

##Getting scripts from link
Using Star-Wars: Next Generation Season 1
"""

! wget 'https://www.st-minutiae.com/resources/scripts/scripts_tng.zip'
! mkdir scripts
! unzip scripts_tng.zip -d scripts


"""##Preprocessing scripts to get dialogues from it"""

! ls -1 scripts > files.txt
with open('files.txt','r') as filestxt:
  files = list(map(lambda x : x.strip(), filestxt.readlines()))
import re
alldialogues = []
```

```
for filename in files:
  with open(f'scripts/{filename}','r',encoding='unicode_escape') as script:
     data = script.readlines()
  dialoguestarts = []
  for idx,i in enumerate(data):
     if (i.isupper() and not i[0].isnumeric() and re.match('                  [\S]',i)):
        dialoguestarts.append(idx)
  dialogues = []
  i = 0
  for i in dialoguestarts:
     dialogue = data[i].strip() + ': '
     line = data[i]
     idx = i
     while line != '\n':
        idx+=1
        line = data[idx]
        dialogue += line.strip() + ' '
     dialogues.append(dialogue)
  alldialogues.extend(dialogues)

"""##Splitting data for test and eval"""
train, test = train_test_split(alldialogues,test_size=0.15)
print(len(train))
print(len(test))
import re
def build_text_files(conversations, dest_path):
    f = open(dest_path, 'w')
    data = ''
    for conv in conversations:
      conv = re.sub(r"<[uU]>|</[uU]>", "", conv)
      data += conv + "  "
    f.write(data)
!mkdir data
build_text_files(train,'/content/data/trainset.txt')
build_text_files(test,'/content/data/testset.txt')

"""##Completing pre-processing needed for GPT"""
tokenizer = GPT2Tokenizer.from_pretrained("gpt2")
train_path = '/content/data/trainset.txt'
test_path = '/content/data/testset.txt'

from transformers import TextDataset,DataCollatorForLanguageModeling

def load_dataset(train_path,test_path,tokenizer):
    train_dataset = TextDataset(
         tokenizer=tokenizer,
         file_path=train_path,
         block_size=192)
    test_dataset = TextDataset(
         tokenizer=tokenizer,
         file_path=test_path,
         block_size=192)
    data_collator = DataCollatorForLanguageModeling(
       tokenizer=tokenizer, mlm=False )
    return train_dataset,test_dataset,data_collator
```

```
train_dataset,test_dataset,data_collator = load_dataset(train_path,test_path,tokenizer)

"""#GPT2 to EnterpriseGPT"""

from transformers import Trainer, TrainingArguments

model = GPT2LMHeadModel.from_pretrained("gpt2")

training_args = TrainingArguments(
    output_dir="content/gdrive/MyDrive/CC/Enterprise",
    overwrite_output_dir=True,
    num_train_epochs=4,
    per_device_train_batch_size=32,
    per_device_eval_batch_size=32,
    eval_steps = 200,
    save_steps=400,
    warmup_steps=250,
    optim='adamw_torch'
    )

trainer = Trainer(
    model=model,
    args=training_args,
    data_collator=data_collator,
    train_dataset=train_dataset,
    eval_dataset=test_dataset,
)

"""##Training Model"""

print("Captain's Logs star date 34592, We're starting the training. Live Long, and
prosper")
trainer.train()
print("Peace and Long Life, Training Complete.")

trainer.save_model()

characters = set()
for i in dialoguestarts:
  characters.add(data[i].strip())
characters

with open('gdrive/MyDrive/CC/Enterprise/charactersSet.txt','w') as charFile:
  for i in characters:
    charFile.write(f'{i}\n')
```

```
"""#Casual Interface for generating dialogues"""

from google.colab import drive
drive.mount('/content/gdrive')

characters = []
with open('gdrive/MyDrive/CC/Enterprise/charactersSet.txt','r') as charFile:
  for i in charFile.readlines():
    characters.append(i.strip())

!pip install transformers gradio
from transformers import pipeline

Enterprise = pipeline('text-generation',model='gdrive/MyDrive/CC/Enterprise/',
tokenizer='gpt2')

result = Enterprise('PICARD (V.O.): ')[0]['generated_text']
result

import gradio as gr
def infer(character, startString):
  return Enterprise(f'{character}:
{startString}',max_new_tokens=120)[0]['generated_text']

casualCreatorApp = gr.Interface(
    fn=infer,
    inputs=[
        gr.Dropdown(
            list(characters), label="Starting of conversation character", info="This is the
character you want the script to talk"
        ),
        gr.Textbox(lines=8,label = "Your Sentence as the character")
        ],
    outputs=[
        gr.Textbox(lines=10,label = "Your GPT generated Star Trek Script")
    ],
)
casualCreatorApp.launch(debug=True)

"""References:

https://github.com/huggingface/transformers/tree/main/examples/pytorch/language-
modeling

https://huggingface.co/docs/transformers

https://www.philschmid.de/fine-tune-a-non-english-gpt-2-model-with-huggingface
"""
```