GPT2_style_transfer

October 10, 2021

1 Mount Google Drive and Install Libraries

[1]:
```
from google.colab import drive
import sys

#Mount your Google drive to the VM
drive.mount('/content/gdrive/')
sys.path.append("/content/gdrive/My Drive/ECE4179/Project")

#set a root path variable to use
ROOT = "/content/gdrive/My Drive/ECE4179/Project/"

#Follow link and give permission, copy code and paste in text box
#You only have to do this once per session
```

```
Mounted at /content/gdrive/
```

[2]:
```
!pip install transformers
!pip install datasets
!pip install nltk
```

```
Collecting transformers
  Downloading transformers-4.11.3-py3-none-any.whl (2.9 MB)
     |                | 2.9 MB 32.2 MB/s
Collecting tokenizers<0.11,>=0.10.1
  Downloading tokenizers-0.10.3-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_6
4.manylinux_2_12_x86_64.manylinux2010_x86_64.whl (3.3 MB)
     |                | 3.3 MB 37.4 MB/s
Requirement already satisfied: tqdm>=4.27 in
/usr/local/lib/python3.7/dist-packages (from transformers) (4.62.3)
Collecting huggingface-hub>=0.0.17
  Downloading huggingface_hub-0.0.19-py3-none-any.whl (56 kB)
     |                | 56 kB 5.0 MB/s
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.7/dist-packages (from transformers) (21.0)
Requirement already satisfied: importlib-metadata in
/usr/local/lib/python3.7/dist-packages (from transformers) (4.8.1)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-
```

```
packages (from transformers) (1.19.5)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-
packages (from transformers) (3.3.0)
Collecting sacremoses
  Downloading sacremoses-0.0.46-py3-none-any.whl (895 kB)
     |                 | 895 kB 42.9 MB/s
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.7/dist-packages (from transformers) (2019.12.20)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-
packages (from transformers) (2.23.0)
Collecting pyyaml>=5.1
  Downloading PyYAML-5.4.1-cp37-cp37m-manylinux1_x86_64.whl (636 kB)
     |                 | 636 kB 46.5 MB/s
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.7/dist-packages (from huggingface-
hub>=0.0.17->transformers) (3.7.4.3)
Requirement already satisfied: pyparsing>=2.0.2 in
/usr/local/lib/python3.7/dist-packages (from packaging>=20.0->transformers)
(2.4.7)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-
packages (from importlib-metadata->transformers) (3.6.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (2021.5.30)
Requirement already satisfied: chardet<4,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (3.0.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (1.24.3)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-
packages (from requests->transformers) (2.10)
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packages
(from sacremoses->transformers) (1.0.1)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages
(from sacremoses->transformers) (1.15.0)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages
(from sacremoses->transformers) (7.1.2)
Installing collected packages: pyyaml, tokenizers, sacremoses, huggingface-hub,
transformers
  Attempting uninstall: pyyaml
    Found existing installation: PyYAML 3.13
    Uninstalling PyYAML-3.13:
      Successfully uninstalled PyYAML-3.13
Successfully installed huggingface-hub-0.0.19 pyyaml-5.4.1 sacremoses-0.0.46
tokenizers-0.10.3 transformers-4.11.3
Collecting datasets
  Downloading datasets-1.12.1-py3-none-any.whl (270 kB)
     |                 | 270 kB 29.2 MB/s
Requirement already satisfied: huggingface-hub<0.1.0,>=0.0.14 in
/usr/local/lib/python3.7/dist-packages (from datasets) (0.0.19)
```

```
Collecting fsspec[http]>=2021.05.0
  Downloading fsspec-2021.10.0-py3-none-any.whl (125 kB)
     |                  | 125 kB 44.5 MB/s
Requirement already satisfied: requests>=2.19.0 in
/usr/local/lib/python3.7/dist-packages (from datasets) (2.23.0)
Requirement already satisfied: pyarrow!=4.0.0,>=1.0.0 in
/usr/local/lib/python3.7/dist-packages (from datasets) (3.0.0)
Collecting xxhash
  Downloading xxhash-2.0.2-cp37-cp37m-manylinux2010_x86_64.whl (243 kB)
     |                  | 243 kB 46.7 MB/s
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-
packages (from datasets) (1.1.5)
Requirement already satisfied: tqdm>=4.62.1 in /usr/local/lib/python3.7/dist-
packages (from datasets) (4.62.3)
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-
packages (from datasets) (21.0)
Collecting aiohttp
  Downloading aiohttp-3.7.4.post0-cp37-cp37m-manylinux2014_x86_64.whl (1.3 MB)
     |                  | 1.3 MB 40.0 MB/s
Requirement already satisfied: multiprocess in
/usr/local/lib/python3.7/dist-packages (from datasets) (0.70.12.2)
Requirement already satisfied: dill in /usr/local/lib/python3.7/dist-packages
(from datasets) (0.3.4)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-
packages (from datasets) (1.19.5)
Requirement already satisfied: importlib-metadata in
/usr/local/lib/python3.7/dist-packages (from datasets) (4.8.1)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.7/dist-packages
(from huggingface-hub<0.1.0,>=0.0.14->datasets) (5.4.1)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.7/dist-packages (from huggingface-
hub<0.1.0,>=0.0.14->datasets) (3.7.4.3)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-
packages (from huggingface-hub<0.1.0,>=0.0.14->datasets) (3.3.0)
Requirement already satisfied: pyparsing>=2.0.2 in
/usr/local/lib/python3.7/dist-packages (from packaging->datasets) (2.4.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.19.0->datasets)
(2021.5.30)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.19.0->datasets)
(1.24.3)
Requirement already satisfied: chardet<4,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.19.0->datasets) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-
packages (from requests>=2.19.0->datasets) (2.10)
Collecting yarl<2.0,>=1.0
  Downloading yarl-1.7.0-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_64.manyl
```

```
inux_2_12_x86_64.manylinux2010_x86_64.whl (271 kB)
     |                  | 271 kB 61.2 MB/s
Requirement already satisfied: attrs>=17.3.0 in
/usr/local/lib/python3.7/dist-packages (from aiohttp->datasets) (21.2.0)
Collecting async-timeout<4.0,>=3.0
  Downloading async_timeout-3.0.1-py3-none-any.whl (8.2 kB)
Collecting multidict<7.0,>=4.5
  Downloading multidict-5.2.0-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_64.
manylinux_2_12_x86_64.manylinux2010_x86_64.whl (160 kB)
     |                  | 160 kB 55.3 MB/s
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-
packages (from importlib-metadata->datasets) (3.6.0)
Requirement already satisfied: python-dateutil>=2.7.3 in
/usr/local/lib/python3.7/dist-packages (from pandas->datasets) (2.8.2)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-
packages (from pandas->datasets) (2018.9)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-
packages (from python-dateutil>=2.7.3->pandas->datasets) (1.15.0)
Installing collected packages: multidict, yarl, async-timeout, fsspec, aiohttp,
xxhash, datasets
Successfully installed aiohttp-3.7.4.post0 async-timeout-3.0.1 datasets-1.12.1
fsspec-2021.10.0 multidict-5.2.0 xxhash-2.0.2 yarl-1.7.0
Requirement already satisfied: nltk in /usr/local/lib/python3.7/dist-packages
(3.2.5)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages
(from nltk) (1.15.0)
```

2 Paraphrasing approach is via reconstruction

- Create dummy dataset; create 'corrupted' Source sentence input/Reference sentence by removing stop word and add noise.
- Then try to reconstruct Reference sentence from source sentence

Resources * https://datachef.co/blog/paraphrasing-with-gpt2/ * https://arxiv.org/pdf/2006.05477.pdf * https://github.com/BH-So/unsupervised-paraphrase-generation * https://huggingface.co/transformers/training.html

[3]:
```
# device = "cuda"
device = "cpu"
max_length = 512
batch_size=2
learning_rate=6.25e-5
num_epochs=5
```

3 Import required libraries

[4]:
```
import numpy as np
from transformers import AutoModelWithLMHead, AutoConfig, Trainer,␣
 ↪AutoTokenizer, TextDataset, DataCollatorForLanguageModeling,␣
 ↪TrainingArguments
import os
from datasets import load_dataset, Dataset
import csv
import logging
import torch
#from torch.utils.data.dataset import Dataset
```

[5]:
```
from transformers import GPT2Tokenizer, GPT2LMHeadModel
import random

import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.tokenize.treebank import TreebankWordTokenizer
from nltk.tokenize.treebank import TreebankWordDetokenizer
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
```

[6]:
```
#set random seeds
np.random.seed(28811666)
torch.manual_seed(28811666)
```

[6]:
```
<torch._C.Generator at 0x7f546fb69170>
```

4 Load text and Create Source sentences from Reference

- Remove stopwords
- Remove 20% of words
- Shuffle 20% of words

[7]:
```
file_name = ROOT + "data/dr_seuss.txt"
#dataset = load_dataset('text', data_files=file_name)
```

[8]:
```
def split_lyrics_file(text_file):
    text = open(text_file, encoding='utf-8').read()
    text = text.split("\n")
    while "" in text:
        text.remove("")
    return text
```

[9]:
```
lines = split_lyrics_file(file_name)
dict_lines = {"lines": lines}
dataset = Dataset.from_dict(dict_lines)
```

[10]:
```
dataset[1]
```

[10]:
```
{'lines': 'By Dr. Seuss'}
```

[11]:
```
tokenizer = TreebankWordTokenizer()
detokenizer = TreebankWordDetokenizer()
english_stopwords = stopwords.words('english')

def remove_stopwords(sentence):
    sentence = tokenizer.tokenize(sentence)
    sentence = [word for word in sentence
                if word.lower() not in english_stopwords]
    sentence = ' '.join(sentence)
    sentence = sentence.replace("''", '"').replace('``', '"')
    sentence = detokenizer.detokenize(sentence.split())
    return sentence

def sentence_noising(sentence, shuffle_ratio=0.2, replace_ratio=0.2):
    # 1. Synonym replacement
    words = sentence.split()
    n_sr = max(1, int(len(words)*shuffle_ratio))
    words = synonym_replacement(words, n_sr)

    # 2. Random shuffling
    if random.random() < shuffle_ratio:
        random.shuffle(words)

    return ' '.join(words)

def data_preparation(in_file, out_file, save_noise_ouput, noised_output,␣
↪max_length):
    gpt_tokenizer = GPT2Tokenizer.from_pretrained('gpt2')
    data = []
    with open(in_file) as f:
        skipped = 0
        for line in f:
            sentence = line.strip()
            corrupted_sentence = remove_stopwords(sentence)
            write_line = corrupted_sentence + '\n' + sentence
            if len(gpt_tokenizer.encode(write_line)) < max_length:
                data.append([corrupted_sentence, sentence])
            else:
                skipped += 1
```

```
    print("Skipped: {}".format(skipped))

    with open(out_file, 'w') as wf:
        writer = csv.writer(wf)
        for corrupted, sentence in data:
            writer.writerow([corrupted, sentence])

    if save_noise_ouput is True:
        with open(noised_output, 'w') as wf:
            writer = csv.writer(wf)
            for corrupted, sentence in data:
                corrupted = sentence_noising(corrupted)
                writer.writerow([corrupted, sentence])

data_preparation(file_name, 'dr_seuss_out.txt', False, 'dr_seuss_out_noised.
↪txt', 100)
```

```
Downloading:   0%|          | 0.00/0.99M [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/446k [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/1.29M [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/665 [00:00<?, ?B/s]

Skipped: 0
```

5 Create Tokenizer

[12]:
```
# tokenizer = AutoTokenizer.from_pretrained('gpt2')
# tokenizer.pad_token = tokenizer.eos_token
```

[13]:
```
def tokenize_function(examples):
    return tokenizer(examples["lines"], padding="max_length", truncation=True)
```

[14]:
```
special_tokens_dict = {'sep_token': '[SEP]'}
tokenizer = GPT2Tokenizer.from_pretrained('gpt2-large')
tokenizer.add_special_tokens(special_tokens_dict)
tokenizer.pad_token = tokenizer.eos_token
```

```
Downloading:   0%|          | 0.00/0.99M [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/446k [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/1.29M [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/666 [00:00<?, ?B/s]
```

[15]:
```
encoded = tokenizer("Hello, I am a single sentence")
print(encoded)
```

```
{'input_ids': [15496, 11, 314, 716, 257, 2060, 6827], 'attention_mask': [1, 1,
1, 1, 1, 1, 1]}
```

6 Create Datasets

[16]:
```python
class TextDataset(Dataset):
    def __init__(self, tokenizer, filename, max_length=512, device='cuda',␣
↪is_inference=False, load_noise_data=False, is_toy=False):
        self.tokenizer=tokenizer
        self.filename = filename
        self.max_length = max_length
        self.device = device
        self.is_inference = bool(is_inference)
        self.is_toy = is_toy

        self.load_dataset(noised=load_noise_data)

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

    def __getitem__(self, idx):
        if torch.is_tensor(idx):
            idx = idx.tolist()
        input_ids = self.input_ids[idx, :].to(self.device)
        samples = {
            'input_ids': input_ids,
        }
        if self.is_inference is False:
            samples['attention_mask'] = \
                    self.attention_mask[idx, :]
            samples['labels'] = self.labels[idx, :]
        return samples

    def load_dataset(self, noised=False):
        filename = self.filename
        if noised is True:
            filename += '.0'
        logging.info("Loading data from {}".format(filename))

        data = []
        with open(filename) as f:
            reader = csv.reader(f)
            for corrupted, sentence in reader:
                data.append([corrupted, sentence])
                if self.is_toy is True:
                    break
```

```
        tokens_list, labels_list = [], []
        for corrupted, sentence in data:
            tokens, labels = self.formatting(corrupted, sentence)
            tokens_list.append(tokens)
            labels_list.append(labels)
        sentences = [self.tokenizer.decode(tokens)
                     for tokens in tokens_list]
        encodings = self.tokenizer(
            sentences, return_tensors='pt', truncation=True,
            padding='max_length', max_length=self.max_length)

        self.input_ids = encodings['input_ids']
        self.attention_mask = encodings['attention_mask']

        if self.is_inference is False:
            self.labels = torch.tensor(labels_list, dtype=torch.long)

    def formatting(self, input_text, target_text):
        input_tokens = self.tokenizer.encode(input_text)
        target_tokens = self.tokenizer.encode(target_text)

        tokens = [self.tokenizer.bos_token_id] + input_tokens \
            + [self.tokenizer.sep_token_id] + target_tokens \
            + [self.tokenizer.eos_token_id]

        labels = [-100] * (len(input_tokens) + 2) \
            + target_tokens + [self.tokenizer.eos_token_id] \
            + [-100] * (self.max_length - len(tokens))
        labels = labels[:self.max_length]
        return tokens, labels
```

[17]:

```
tokenized_datasets = dataset.map(tokenize_function, batched=True)
```

```
  0%|          | 0/2 [00:00<?, ?ba/s]
```

[18]:

```
print(tokenized_datasets["lines"][2])
print(tokenized_datasets["input_ids"][2])
```

```
The sun did not shine.
[464, 4252, 750, 407, 18340, 13, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
```

50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,

```
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256,
50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256, 50256]
```

[19]:
```
small_train_dataset = tokenized_datasets.shuffle(seed=42).select(range(100))
full_train_dataset = tokenized_datasets
```

7 Define Model

[20]:
```
model = GPT2LMHeadModel.from_pretrained('gpt2-large', pad_token_id = tokenizer.
↪eos_token_id)
model.resize_token_embeddings(len(tokenizer))
```

```
model
```

```
Downloading:   0%|          | 0.00/3.02G [00:00<?, ?B/s]
```

[20]:
```
GPT2LMHeadModel(
  (transformer): GPT2Model(
    (wte): Embedding(50258, 1280)
    (wpe): Embedding(1024, 1280)
    (drop): Dropout(p=0.1, inplace=False)
    (h): ModuleList(
      (0): GPT2Block(
        (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
        (attn): GPT2Attention(
          (c_attn): Conv1D()
          (c_proj): Conv1D()
          (attn_dropout): Dropout(p=0.1, inplace=False)
          (resid_dropout): Dropout(p=0.1, inplace=False)
        )
        (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
        (mlp): GPT2MLP(
          (c_fc): Conv1D()
          (c_proj): Conv1D()
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
      (1): GPT2Block(
        (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
        (attn): GPT2Attention(
          (c_attn): Conv1D()
          (c_proj): Conv1D()
          (attn_dropout): Dropout(p=0.1, inplace=False)
          (resid_dropout): Dropout(p=0.1, inplace=False)
        )
        (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
        (mlp): GPT2MLP(
          (c_fc): Conv1D()
          (c_proj): Conv1D()
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
      (2): GPT2Block(
        (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
        (attn): GPT2Attention(
          (c_attn): Conv1D()
          (c_proj): Conv1D()
          (attn_dropout): Dropout(p=0.1, inplace=False)
          (resid_dropout): Dropout(p=0.1, inplace=False)
```

```
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(3): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(4): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(5): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
```

```
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(6): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(7): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(8): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
```

```
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(9): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(10): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(11): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
```

```
  )
)
(12): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(13): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(14): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
```

```
(15): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(16): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(17): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(18): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
```

```
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(19): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(20): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(21): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
```

```
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(22): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(23): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(24): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
```

```
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(25): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(26): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(27): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
```

```
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(28): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(29): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(30): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
```

```
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(31): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(32): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(33): GPT2Block(
  (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (attn): GPT2Attention(
    (c_attn): Conv1D()
    (c_proj): Conv1D()
    (attn_dropout): Dropout(p=0.1, inplace=False)
    (resid_dropout): Dropout(p=0.1, inplace=False)
  )
  (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
  (mlp): GPT2MLP(
    (c_fc): Conv1D()
    (c_proj): Conv1D()
```

```
        (dropout): Dropout(p=0.1, inplace=False)
      )
    )
    (34): GPT2Block(
      (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
      (attn): GPT2Attention(
        (c_attn): Conv1D()
        (c_proj): Conv1D()
        (attn_dropout): Dropout(p=0.1, inplace=False)
        (resid_dropout): Dropout(p=0.1, inplace=False)
      )
      (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
      (mlp): GPT2MLP(
        (c_fc): Conv1D()
        (c_proj): Conv1D()
        (dropout): Dropout(p=0.1, inplace=False)
      )
    )
    (35): GPT2Block(
      (ln_1): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
      (attn): GPT2Attention(
        (c_attn): Conv1D()
        (c_proj): Conv1D()
        (attn_dropout): Dropout(p=0.1, inplace=False)
        (resid_dropout): Dropout(p=0.1, inplace=False)
      )
      (ln_2): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
      (mlp): GPT2MLP(
        (c_fc): Conv1D()
        (c_proj): Conv1D()
        (dropout): Dropout(p=0.1, inplace=False)
      )
    )
  )
  (ln_f): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
)
(lm_head): Linear(in_features=1280, out_features=50258, bias=False)
)
```

8 Fine-tune model

- Use pre-trained GPT-2 checkpoint from HuggingFace Library
- Fine-tune model on reconstruction task
- Concatenate Source and Reference sequence separated by special symbol to form input sequence

[61]:
```
# Fine-tune with author text data
training_args = TrainingArguments("test_trainer", num_train_epochs=num_epochs,␣
 ↪per_device_train_batch_size=batch_size,
                                  learning_rate=learning_rate, weight_decay=0.
 ↪01, seed=28811666)
trainer = Trainer(
    model=model, args=training_args, train_dataset=small_train_dataset,␣
 ↪eval_dataset=small_train_dataset
)
#trainer.train()
```

9 Test encoder and decoder

[62]:
```
# Encode text and return torch tensors 'pt' (PyTorch tensors)
# Converts words into numbers (indices)
topic = "The weather is nice"
input_ids = tokenizer.encode(topic, return_tensors = 'pt')
input_ids
```

[62]:
```
tensor([[ 464, 6193,  318, 3621]])
```

[63]:
```
# Decode text from indices
tokenizer.decode(input_ids[0])
```

[63]:
```
'The weather is nice'
```

10 Generate Text

10.1 Beam Search text generation

[64]:
```
# Generate text using generate function from GPT2LMHeadModel via beam search
# https://huggingface.co/blog/how-to-generate
# Args: max_length: maximum number of words in generated text
#       num_beams: beam search reduces risk of missing hidden high probability␣
 ↪word sequences by keeping the most
#                  likely num_beams of hypotheses at each time step and␣
 ↪eventually choosing the hypothesis that has
#                  the overall highest probability
#       no_repeat_ngram_size: while result is arguably more fluent, output␣
 ↪still includes repetition of same word seqs
#                  introduce n-grams (word seqs of n words) penalties
output_beam = model.generate(input_ids, max_length = 500, num_beams = 5,
                        no_repeat_ngram_size  = 2, early_stopping = True)
```

[65]:
```
print(tokenizer.decode(output_beam[0], skip_special_tokens = True))
```

```
The weather is nice today, but it's going to be a long day."

"Yeah, I know. I'm just glad we're not in the middle of a snowstorm. It would
have been a lot worse if we were in a blizzard or something like that. We could
have gotten stuck in there for a while, and I don't think we'd be able to get
out of there. But I guess we'll just have to wait and see what the weather's
like tomorrow, then we can decide whether or not we want to go back out there."
She smiled at me. "I'm glad you're okay with it, though. You're the first person
I've told about this, after all." I smiled back at her. She was a good friend,
even if she wasn't always the best person to talk to about things like this.
Maybe that was why I liked her so much, because she was always willing to listen
to me and try to help me out when I was in trouble. That was something I really
appreciated about her, in addition to the fact that she always seemed to know
what was going on in my head and what I needed to do to make things better. Even
though she didn't know everything, she knew enough to give me advice and make me
feel better about myself. And that made me really happy, especially when she
would tell me things that I would never have thought of on my own, like how I
shouldn't be afraid of the dark, or that it was okay to cry when you were sad.
Or that if you had a crush on someone, you should just let it go and move on
with your life, instead of trying to force it on them. Those were the kinds of
things she told me all the time, which made it really easy for me to believe in
her and her advice. When I told her about how much I wanted to see her again, it
made her smile even wider, as if to say, "Oh, that's so sweet of you to think of
me that way. Of course I'd want you back, of course you would want me back." And
then she hugged me tightly and kissed me on the cheek, saying "Goodbye, sweetie.
See you tomorrow." Then she turned around and walked away, leaving me alone with
my thoughts. After a few minutes of thinking about what she had just said, my
mind started to wander back to what had happened earlier.
```

10.2 Top-K sampling text generation

[66]:
```
# Generate text using generate function from GPT2LMHeadModel via Top-K sampling
# K most likely next words are filtered and probability mass is redistributed␣
 ↪among only those K next words
# Method adopted by GPT2
output_topk = model.generate(input_ids, do_sample=True, max_length = 500,␣
 ↪top_k=50)
```

[67]:
```
print(tokenizer.decode(output_topk[0], skip_special_tokens = True))
```

```
The weather is nice. We had a lovely walk on the beach at midnight and at about
3 a.m. when the moon came up I knew you were going to be here. It's a beautiful
day. It's sunny outside. It's a beautiful day for swimming and snorkeling."

On Thursday, he made a second surprise visit.

"He went to go buy a suit," said Kelly.
```

```
And he had planned it since May, too, when he got together with Katt Williams
and two other friends.

"He said, 'All right everyone - let's go.'"

That first day, the group rode down from California in a van. One took in the
view.

"We just rode around," Wills said. "He came out of nowhere as a black van."

They met people who asked where they had been and if their van had been stolen.

When his friend wanted some water, Wills offered. They drove through the Santa
Susana Wilderness Park and picked a few places where they wouldn't be seen.

One thing was for sure, Wills never imagined the day would come when Katt would
have his own beach. They were in the car - he was sitting, she was sitting by
the trunk of the van - when they heard a loud knock at the door.

"I said, 'Yes, I'll come in.' And we put the van in park."

They'd gone on a date.

"I am still friends with him."

- - -

There is something about San Diego that makes you want to feel like a part of
it, whether you're from this community or someone else. In this region with so
many things to do and so much to see, it's easy to become forgettable. But I
love San Diego, too, and that's why it's difficult to leave.

We have so much history here. From the first European settlers to the people who
built and built and built, we built our modern day life here. From the first
great city blocks and skyscrapers, to the first skyscrapers to be torn down or
redesigned, to the first office towers to be demolished, we built a great city
where people know their neighbors and take pride in their community.

But we also have so much more to learn about the new and better ways of seeing
our city
```

10.3 Top-p sampling text generation

[68]:
```
# Generate text using generate function from GPT2LMHeadModel via Top-p sampling
# Aka nucleus-sampling; top-p sampling chooses the smallest possible set of␣
 ↪words whose cumulative property
```

```
# exceeds the probability p; probability mass is redistributed among this set␣
 ↪of words
# This means, the size od the set of words can be dynamically increased and␣
 ↪decreased according to
# next word's probability distribution
output_topp = model.generate(input_ids, do_sample=True, max_length=500, top_p=0.
 ↪92, top_k=0)
```

[69]:
```
print(tokenizer.decode(output_topp[0], skip_special_tokens = True))
```

```
The weather is nice, so we're going to try to do the best we can to make sure
our prospects are good enough to try to sign him.

"He's obviously a lovely player, he's not just a good player, he's a superb one
as well."

The Red Bulls' latest signing for next season was goalkeeper Sean Johnson,
acquired from Colorado on Monday in exchange for the free-agent Matt Pickens, as
part of the club's turnaround under head coach Jesse Marsch.

Lenz always played for the Red Bulls Under-23 team, playing in all of their MLS
Reserve League matches, as well as having a brief run in the last three CONCACAF
Champions League matches.

As part of the transaction, SKC did not have a player pick up his Designated
Player option, but will have up to $800,000 in allocation money should they
exercise it.
```

10.4 Using both Top-k and Top-p in text generation

[70]:
```
# Both work well in practice, so lets use both toegher to avoid very low ranked␣
 ↪words while allowing for some dynamic selection
output_topkp = model.generate(input_ids, do_sample=True, max_length=500,␣
 ↪top_k=50, top_p=0.95)
```

[71]:
```
print(tokenizer.decode(output_topkp[0], skip_special_tokens = True))
```

```
The weather is nice and nice today! Just perfect for a warm start to the year.


I got home at about 9:00pm this afternoon to find a package waiting for me! The
first thing I noticed was that it was a huge box and I wasn't quite sure how it
was made (no picture on Amazon?!?!) but when I opened the box I saw that it
wasn't just a package, but a giant treat… Mascarpone! It looked like it was a
lot larger than a regular Mascarpone, and I'm already excited!
```

I thought that it would be awesome to try out a fresh one before the holidays, so I went home, and proceeded to eat most of it. I'm not sure what to expect with the Mascarpone so far, but I am looking forward to it. It is definitely the first time I have had Mascarpone, and I am curious to see how it compares to my regular Mascarpone! It sounds good so far!

10.5 Generate more text

[72]:
```
# Encode text and return torch tensors 'pt' (PyTorch tensors)
# Converts words into numbers (indices)
topic = "Neural networks and deep learning"
input_ids = tokenizer.encode(topic, return_tensors = 'pt')
input_ids
```

[72]:
```
tensor([[8199, 1523, 7686,  290, 2769, 4673]])
```

[74]:
```
# Beam search
output_beam = model.generate(input_ids, max_length = 500, num_beams = 5,
                        no_repeat_ngram_size  = 2, early_stopping = True)
print(tokenizer.decode(output_beam[0], skip_special_tokens = True))
```

```
Neural networks and deep learning

Deep learning is a type of machine learning that uses deep neural networks
(DNNs) to learn from large amounts of data. It is based on the idea that the
more data you have, the better you will be able to understand the data and make
predictions about it. Deep learning can be applied to a wide range of problems,
including image recognition, speech recognition and natural language processing
(NLP). In this article, we will look at some of the most common applications of
DNN and how they are being used in the real world. We will also discuss the
different types of Deep Learning algorithms and their strengths and weaknesses,
as well as how you can use them in your own applications.
```

[75]:
```
# Top-k
output_topk = model.generate(input_ids, do_sample=True, max_length = 500,␣
 ↪top_k=50)
print(tokenizer.decode(output_topk[0], skip_special_tokens = True))
```

```
Neural networks and deep learning. In Proceedings of the ACM Computer Vision and
Pattern Recognition (CVPR) Congress 2017[1],

http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2698320

https://medium.com/@kim_powell/deep-learning-and-fetch-in-a-sensor-
neuralnet-8c934f58cab8#.gwv2lmnx7

http://arxiv.org/abs/1403.02255
```

http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2698331

http://arxiv.org/abs/1403.08014

http://arxiv.org/abs/1407.02050

http://arxiv.org/abs/1409.03634

http://arxiv.org/abs/1407.03089

http://arxiv.org/abs/1407.03153

http://arxiv.org/abs/1407.03157

http://arxiv.org/abs/1403.09041

http://arxiv.org/abs/1403.09041

https://arxiv.org/abs/1403.09084

B.D. Wong, H.Y. Lee, J. Wu, T.T. Nguyen and J.R. Nguyen. 2016. Neural Machine Learning with Deep Feature Selection. arXiv:1511.04759[abstract]

ArXiv:1611.06822[abstract]

2016. A Deep Learning Network that Understands Text Image Representations: Learning from Input Images to Predict With a Convolutional Neural Network. arXiv:1612.04757[abstract]

ArXiv:1948.04903[abstract]

2016. LSTM Deep Learning Architecture for Image Recognition with Deep Convolutional Neural Network. arXiv:1708.04747[abstract]

Abstract: We propose a generalized deep learning architecture for the classifiers of image classification tasks. Our architecture is based on a LSTM,

[76]:
```
# Top-p
output_topp = model.generate(input_ids, do_sample=True, max_length=500, top_p=0.
 ↪92, top_k=0)
print(tokenizer.decode(output_topp[0], skip_special_tokens = True))
```

Neural networks and deep learning - insights from mathematical and computer science expertise to solve real problems

Ravi Raghu, Tetsuya Ishikawa, Kenji Miyamoto

Kinnogo, Japan: Interactions of dynamical and stochastic systems and

architecture

Takashi, Yoshinori Ohshima, Naoya Ohtsu

Masato Nakamura, Makoto Ishimoto, Naoya Takeuchi

Tokyo, Japan: (As of 25 Oct 2018)

Personalization in Online advertisements

Mitsuhiro Nakamura, Masato Nakamura

Minato, Japan: (As of 28 Oct 2018)

Distributed subnets in the Internet

David Fish, Bill Habib, Mitosh Kumar, Mu-Kwon Lee, Haumea Abd-Alghani, Peter Antonopoulos, Andres Roque, Garth Twal, Mark Tumminio, Patrick Wu

Tokyo, Japan: (As of 14 Nov 2018)

Mechanisms of Persistent Patterns in Medical Databases

Martin Ito, Paolo Paparella, Danilo Arcimboldo, Jorge R. Seveso

3-4, UK: (As of 12 Nov 2018)

Attentional Bias in Great Works of Art

Sascha Lips, Richard Gillespie, C. Hassabis Andrade

Thessaloniki, Greece: (As of 10 Nov 2018)

Adherence Detection and Response in Complex Multiple Choice Games

Mark Rudolph, Simon Purdy, Alexander Berg, Kai Lin, Martin Sees

Daegu, Korea: (As of 28 Oct 2018)

You are all the best! - rate that quote

Chris Woodruff, Lee Kobori, Ricky Lee, Michiko Miura, Naoko Kihara

Niigata, Japan: (As of 24 Oct 2018)

Stochastic Optimization: Tuning Artificial Neural Networks to Improve Performance

Arian Sarafian, Mario Spatafora, Isaac Ruiz-Sanchez, Jaroslav Stanko

Université de Strasbourg, France: (As of 7 Nov 2018)

Stochastic Optimization: Tuning Artificial Neural Networks to Improve Performance

Claire Brooks, Kevin G. Carr, Torbjørn Lodergaard, Paolo Padovani

[77]:
```
# Top-k and Top-p
output_topkp = model.generate(input_ids, do_sample=True, max_length=500,␣
 ↪top_k=50, top_p=0.95)
print(tokenizer.decode(output_topkp[0], skip_special_tokens = True))
```

Neural networks and deep learning systems are often used for a variety of tasks such as image recognition, speech recognition, natural language processing, and the detection of medical anomalies and other health conditions. The current research in the field of visual speech recognition has generated major progress in the way that this technology is being used, with applications ranging from detection of impaired vision, to speech perception and control, to computer translation of spoken language. While some of the current research may seem somewhat esoteric, there is a compelling reason to use this type of technology.

As the brain is the source of language, it is interesting that a system that can interpret your speech is also considered the source of your words. This is a major advantage of computer vision because the computer can learn from your visual experience and apply this to the meaning of the words your brain is trying to interpret. This allows a computer system to make improvements in its performance.

One example of a system that attempts to emulate natural speech would be an artificial brain that is trained by looking at human speech recordings. This artificial brain, or a human brain, is able to learn a language by observing the expressions, tone, and pauses that humans do, so it can imitate the expressions and tone of natural human speech. To accomplish this, the system would have to be modeled based on the behavior of the brain's neurons and neural networks, and has to be able to learn how human language works.

Deep Learning and Machine Learning

The following list of research areas contains a collection of computer-based systems that are aimed at solving various tasks associated with a variety of fields, such as image analysis, natural language processing, speech recognition, image processing, sound generation, and video visualization.

The research in deep learning has produced some of the biggest technological advances in the last ten years, and can even assist in image and video

production. There is a plethora of research in this area, from the fundamental aspects of deep learning, to deep learning systems that can help produce professional quality videos.

Artificial neural networks have been found to produce images faster and more accurately than traditional networks that process digital information, and have helped to revolutionize machine learning, particularly in the fields of image processing, speech recognition, and natural language processing. AI systems have been shown to perform better and more accurately than humans in certain image recognition tasks, such as recognizing faces, and have also been found to outperform humans at performing sound synthesis, image synthesis, and speech

10.6 Generate more text

[78]:
```
# Encode text and return torch tensors 'pt' (PyTorch tensors)
# Converts words into numbers (indices)
topic = "Covid vaccinations in Australia"
input_ids = tokenizer.encode(topic, return_tensors = 'pt')
input_ids
```

[78]:
```
tensor([[   34,   709,   312, 46419,   287,  4505]])
```

[79]:
```
# Beam search
output_beam = model.generate(input_ids, max_length = 500, num_beams = 5,
                        no_repeat_ngram_size  = 2, early_stopping = True)
print(tokenizer.decode(output_beam[0], skip_special_tokens = True))
```

```
Covid vaccinations in Australia and New Zealand

The Australian Vaccination Network (AVN) is an independent, non-profit
organisation that works to ensure that all children are vaccinated against
diphtheria, tetanus, pertussis (whooping cough), polio, measles, mumps and
rubella (MMR) and Haemophilus influenzae type b (Hib) vaccines. The AVN is a
member of the World Health Organisation (WHO) Vaccine Advisory Committee (VAC),
which is responsible for advising the WHO on the safety and efficacy of all
vaccines, including those for which there is no evidence of safety or efficacy
in humans. In addition, the VAC advises the Australian Government on vaccination
policy and provides advice to the public on immunisation. For more information,
please visit www.avn.org.au.
```

[80]:
```
# Top-k
output_topk = model.generate(input_ids, do_sample=True, max_length = 500,␣
 ↪top_k=50)
print(tokenizer.decode(output_topk[0], skip_special_tokens = True))
```

```
Covid vaccinations in Australia are already part of the standard vaccination for
Australian schools.
```

'People are terrified of getting this vaccine because it's too dangerous,' said Mr Choudhary.

A group of parents protested a vaccination day in Sydney, Australia, on Saturday while in support of the right of parents to decide whether their children should be vaccinated against deadly childhood diseases

Choudhary said the group will go ahead with protests in London.

'We need that vaccination against all diseases,' he told the Daily Mail, adding that he hoped the protest would lead to a debate about vaccinations.

'It should not be against religion and a belief, but the right of the freedom of people to do religion and have a belief that should be respected as much as possible,' he said.

The issue of vaccinations has become controversial in Britain when it is estimated that as many as 1 in 10 children are at risk of contracting dangerous diseases such as measles - a disease which is extremely rare in developed countries, but is particularly dangerous for young children who are too young to be vaccinated.

[81]:
```
# Top-p
output_topp = model.generate(input_ids, do_sample=True, max_length=500, top_p=0.
 ↪92, top_k=0)
print(tokenizer.decode(output_topp[0], skip_special_tokens = True))
```

Covid vaccinations in Australia.

PEP-II is an immunisation regime comprising a vaccine against HPV type 16 or 18, mixed vaccination of HCMV and Acanthamoeba histolytica (AH) with HCMV-Y strains I to VI and a chickenpox virus vaccine, which is a live-attenuated

strain from the HCMV serotype 16 homolog K and which may have been used in some vaccine trials to protect

against infection with a varicella vaccine-type varicella (VV) virus.

It has been extensively tested in the USA, France, Sweden and New Zealand with efficacy rates between 60% and 75% for HCMV-Y, with efficacy rates ranging between 10% and 22% for AH.

We assume that 7 out of 9 children should receive the PEP II vaccines for HPV16/18 and for AH.

It is recommended that every child aged 12 months with a medical condition have

a health history form within 48 hours and be referred for standard laboratory testing of salivary salivary secretions in order to screen for diseases, before the PEP vaccines are administered. Also, mother's blood should be tested in order to ensure optimal care and attention to maternal health.

Evidence-based clinical practice

We use 13 clusters of previously published clinical practice observations (CLOs) for the guideline practice of caring for HPV16/18, HPV18 and other HPV-related genital infections. The number of clusters grouped by patient interest and clinical practice were used as a measure of consensus (clinician-observation).

Schizophrenia and mental disorders (psychosocial, autonomic and behavioural) may be co-morbid with certain genital warts. In 2013, 394 children diagnosed with schizophrenia-spectrum disorders who were 12 years old or older in 2009 were examined at the University of Toronto, Canada. 397 of these children were referred to a neuropsychiatry clinic in order to have diagnostic neuropsychiatric surgery to remove the warts. Only 27 of these children (10.3%) received an HPV16/18 vaccine. In a 2014 report, 40.4% of children from the same clinic who received a second dose of an HPV16/18 vaccine had improved response to treatment with antisense oligonucleotide, 20.5% of these with improvement to drugs. A published 2005

[82]:
```
# Top-k and Top-p
output_topkp = model.generate(input_ids, do_sample=True, max_length=500,␣
 ↪top_k=50, top_p=0.95)
print(tokenizer.decode(output_topkp[0], skip_special_tokens = True))
```

Covid vaccinations in Australia and the United Kingdom have proven to be very effective in eradicating smallpox. The vaccine can only be used if there are no known cases of infection. If smallpox were to return it could be fatal.

The main danger from smallpox is not its appearance in person, but its spread and its transmission in people. Vaccination has been proven effective and safe in protecting against smallpox.

Since the World War II smallpox infections in children in the U.S. have almost completely ceased. There have been cases from time to time of vaccine-derived smallpox and in the U.S. have been linked to the vaccination of U.S. soldiers who were exposed to Soviet soldiers during the Cold War. The incidence of smallpox has remained under control in this country for nearly 50 years.

Dr. E. Michael Griffin is the editor of Vaccine Times and has written many science columns and books in the area. He also consults for private insurance companies, the U.S. Department of Veterans Affairs and the U.S. Defense Department.