In [2]:
```
!pip install transformers
!pip install wandb
!pip install trl
!pip install pandas
!pip install datasets
!pip install accelerate
!pip install tyro
!pip install nltk -U
```

```
ll20/apps/anaconda/2020.11-py38-gcc-4.8.5-djkvkvk/lib/python3.8/site-packag
es (from wandb) (7.1.2)
Requirement already satisfied: psutil>=5.0.0 in /apps/spack/scholar/fall20/
apps/anaconda/2020.11-py38-gcc-4.8.5-djkvkvk/lib/python3.8/site-packages (f
rom wandb) (5.7.2)
Requirement already satisfied: setproctitle in ./.local/lib/python3.8/site-
packages (from wandb) (1.3.3)
Requirement already satisfied: PyYAML in /apps/spack/scholar/fall20/apps/an
aconda/2020.11-py38-gcc-4.8.5-djkvkvk/lib/python3.8/site-packages (from wan
db) (5.3.1)
Requirement already satisfied: appdirs>=1.4.3 in ./.local/lib/python3.8/sit
e-packages (from wandb) (1.4.4)
Requirement already satisfied: setuptools in /apps/spack/scholar/fall20/app
s/anaconda/2020.11-py38-gcc-4.8.5-djkvkvk/lib/python3.8/site-packages (from
wandb) (50.3.1.post20201107)
Requirement already satisfied: protobuf!=4.21.0,<5,>=3.12.0; python_version
< "3.9" and sys_platform == "linux" in ./.local/lib/python3.8/site-packages
(from wandb) (4.25.3)
Requirement already satisfied: GitPython!=3.1.29,>=1.0.0 in ./.local/lib/py
thon3.8/site-packages (from wandb) (3.1.42)
```

In [3]:
```
import torch
from tqdm import tqdm
import pandas as pd
import wandb
import os

tqdm.pandas()

from transformers import pipeline, AutoTokenizer
from datasets import load_dataset

from trl import PPOTrainer, PPOConfig, AutoModelForCausalLMWithValueHead
from trl.core import LengthSampler
```

In [4]:
```
config = PPOConfig(
    model_name    = "mrm8488/distilroberta-finetuned-financial-news-sentiment-a
    learning_rate = 1.41e-5,
    ## log_with      = "wandb",
)

sent_kwargs = {
         "return_all_scores": True,
         "function_to_apply": "none",
         "batch_size": 16
}
```

In [5]:
```
## wandb.init()

wandb.init(mode="disabled")
os.environ['WANDB_DISABLED'] = 'true'
```

```
Failed to detect the name of this notebook, you can set it manually with the
WANDB_NOTEBOOK_NAME environment variable to enable code saving.
```

Load 'financial_phrasebank' dataset

Polar sentiment dataset of sentences from financial news. The dataset consists of 4840 sentences from English language financial news categorised by sentiment. The dataset is divided by agreement rate of 5-8 annotators.

Visualize details of dataset

In [6]:
```
dataset_name="financial_phrasebank"
```

In [7]:
```
ds  = load_dataset(dataset_name,'sentences_50agree', split = "train")
```

In [8]:
```
ds
```

Out[8]:
```
Dataset({
    features: ['sentence', 'label'],
    num_rows: 4846
})
```

In [9]:
```
ds[15:18]
```

Out[9]:
```
{'sentence': ['Consolidated net sales increased 16 % to reach EUR74 .8 m , wh
ile operating profit amounted to EUR0 .9 m compared to a loss of EUR0 .7 m in
the prior year period .',
  'Foundries division reports its sales increased by 9.7 % to EUR 63.1 mn fro
m EUR 57.5 mn in the corresponding period in 2006 , and sales of the Machine
Shop division increased by 16.4 % to EUR 41.2 mn from EUR 35.4 mn in the corr
esponding period in 2006 .',
  "HELSINKI ( AFX ) - Shares closed higher , led by Nokia after it announced
plans to team up with Sanyo to manufacture 3G handsets , and by Nokian Tyres
after its fourth-quarter earnings report beat analysts ' expectations , deale
rs said ."],
 'label': [2, 2, 2]}
```

In [10]:
```
from datasets import ClassLabel
import random
import pandas as pd
from IPython.display import display, HTML
```

In [11]:
```
def show_random_elements(dataset, num_examples=20):
    assert num_examples <= len(dataset), "Can't pick more elements than there a

    picks = []

    for _ in range( num_examples ):

        pick = random.randint(0, len(dataset)-1)
        while pick in picks:
            pick = random.randint(0, len(dataset)-1)
        picks.append(pick)

    df = pd.DataFrame( dataset[picks] )        ## indexing 10 picks

    print(df)
    print(dataset.features.items())

    for column, typ in dataset.features.items():
        print(column)
        print(typ)
        print(ClassLabel)
        ## The isinstance() function returns True if the specified object
        ## is of the specified type, otherwise False
        if isinstance(typ, ClassLabel):
            print("Hello")
            df[column] = df[column].transform(lambda i: typ.names[i])
            ## print(typ.names[i])

    display(HTML(df.to_html()))
```

In [12]:
```
show_random_elements(ds)
```

```
                                              sentence  label
0   Incap Corporation Stock Exchange Announcement ...      1
1   Finnish Bank of +àland reports its operating p...      0
2   Stichting Pensioenfonds ABP : 4 118 122 shares...      1
3   Simultaneously , Alma Media has purchased a 35...      1
4   Profit after taxes was EUR 0.1 mn , compared t...      2
5   Glaston , headquartered in Tampere , Finland ,...      2
6   The hull of the vessel was built one block at ...      1
7   He also mentions that this improvement to the ...      2
8   Finnish real estate investor Sponda Plc said o...      1
9   Protalix is developing genetically engineered ...      1
10  The 5,000 megawatt wind farm being planned in ...      1
11  She will succeed Krister Kylas , who has decid...      1
12  The other seats would go to Edgar Edmonds , an...      1
13  Progress Group , QPR 's representative in Saud...      2
14  Tallink claims the watertight doors of both Va...      2
15  The above mentioned shareholders will suggest ...      1
16  Mr Skogster currently serves as the manager re...      1
17  The company feels these leases are prime locat...      2
18  Finnish power supply solutions and systems pro...      0
19  Earnings per share ( EPS ) were EUR0 .03 , up ...      2
dict_items([('sentence', Value(dtype='string', id=None)), ('label', ClassLabe
l(names=['negative', 'neutral', 'positive'], id=None))])
sentence
Value(dtype='string', id=None)
<class 'datasets.features.features.ClassLabel'>
label
ClassLabel(names=['negative', 'neutral', 'positive'], id=None)
<class 'datasets.features.features.ClassLabel'>
Hello
```

	sentence	lab
0	Incap Corporation Stock Exchange Announcement 29 April 2010 at 1 p.m. INVITATION TO A NEWS CONFERENCE ON INCAP 'S INTERIM REPORT Q1-2010 Incap will publish its interim report for January-March 2010 on Wednesday , 5 May 2010 .	neut
1	Finnish Bank of +àland reports its operating profit fell to EUR 4.9 mn in the third quarter of 2007 from EUR 5.6 mn in the third quarter of 2006 .	negati
2	Stichting Pensioenfonds ABP : 4 118 122 shares representing 5.19 % of the share capital and voting rights .	neut
3	Simultaneously , Alma Media has purchased a 35 % share of Arena Interactive , a subsidiary of Arena Partners with a focus on mobile solutions development .	neut
4	Profit after taxes was EUR 0.1 mn , compared to EUR -0.4 mn the previous year .	positi
5	Glaston , headquartered in Tampere , Finland , is a growing , international glass technology company .	positi
6	The hull of the vessel was built one block at a time and Ruukki delivered the plate material for each block as construction progressed .	neut
7	He also mentions that this improvement to the service follows the recent expansion of the Finnlines service from Bilbao via Antwerp and Helsinki and from Hull via Helsinki to St. Petersburg .	positi
8	Finnish real estate investor Sponda Plc said on Wednesday 12 March that it has signed agreements with Danske Bank A-S , Helsinki Branch for a 7-year EUR150m credit facility and with Ilmarinen Mutual Pension Insurance Company for a 7-year EUR50m credit facility .	neut
9	Protalix is developing genetically engineered proteins from plant cells .	neut
10	The 5,000 megawatt wind farm being planned in Raahe would be built offshore in front of Ruukki 's Raahe Works .	neut
11	She will succeed Krister Kylas , who has decided to leave TeliaSonera .	neut
12	The other seats would go to Edgar Edmonds , an American with experience of the clothing and retail industry , and Christian Fischer , an Austrian with experience in the winter sports goods business .	neut
13	Progress Group , QPR 's representative in Saudi Arabia and North Africa , has signed a framework agreement for a long term strategic relationship with ISE .	positi
14	Tallink claims the watertight doors of both Vana Tallinn and Regina Baltica , including their electrical systems , are fully in working order .	positi
15	The above mentioned shareholders will suggest that a monthly salary of EUR 1,400 would be paid for the Board members outside the company .	neut
16	Mr Skogster currently serves as the manager responsible for ABB Oy 's system modules for low voltage drives .	neut
17	The company feels these leases are prime locations due to several producing formations in the immediate area .	positi
18	Finnish power supply solutions and systems provider Efore Oyj said its net loss widened to 3.2 mln euro $4.2 mln for the first quarter of fiscal 2006 - 2007 ending October 31, 2007 from 900,000 euro$ 1.2 mln for the same period of fiscal 2005-06 .	negati
19	Earnings per share (EPS) were EUR0 .03 , up from the loss of EUR0 .083 .	positi

In [13]:
```
# ds = ds.rename_columns({"text": "sentence"})
ds = ds.filter(lambda x: len(x["sentence"]) > 200, batched=False)
```

In [14]:
```
ds
```

Out[14]:
```
Dataset({
    features: ['sentence', 'label'],
    num_rows: 625
})
```

In [15]:
```
tokenizer           = AutoTokenizer.from_pretrained(config.model_name)
tokenizer.pad_token = tokenizer.eos_token
```

In [16]:
```
def tokenize( sample ):
    sample["input_ids"] = tokenizer.encode( sample["sentence"]    )[: 20]
    sample["query"]     = tokenizer.decode( sample["input_ids"] )
    return sample


ds = ds.map(tokenize, batched=False)
ds
```

Out[16]:
```
Dataset({
    features: ['sentence', 'label', 'input_ids', 'query'],
    num_rows: 625
})
```

In [17]:
```
ds[15:18]
```

Out[17]:

```
{'sentence': ["Seppala 's revenue increased by 0.2 % to EUR10 .1 m. In Finlan
d , revenue went down by 2.4 % to EUR6 .8 m , while sales abroad rose by 6.2
% to EUR3 .3 m. Sales increased in all the Baltic countries as well as in Rus
sia and Ukraine .",
  "At the request of Finnish media company Alma Media 's newspapers , researc
h manager Jari Kaivo-oja at the Finland Futures Research Centre at the Turku
School of Economics has drawn up a future scenario for Finland 's national ec
onomy by using a model developed by the University of Denver .",
  "`` The new agreement is a continuation to theagreement signed earlier this
year with the Lemminkainen Group , whereby Cramo acquired the entire construc
tion machine fleet ofLemminkainen Talo Oy Ita - ja Pohjois Suomo , and signed
asimilar agreement , '' said Tatu Hauhio , managing director ofCramo Finland
."],
 'label': [2, 1, 1],
 'input_ids': [[0,
   14696,
   3807,
   2331,
   128,
   29,
   903,
   1130,
   30,
   321,
   4,
   176,
   7606,
   7,
   10353,
   698,
   479,
   134,
   475,
   4],
  [0,
   3750,
   5,
   2069,
   9,
   21533,
   433,
   138,
   33277,
   2454,
   128,
   29,
   9911,
   2156,
   557,
   1044,
   344,
   1512,
   7916,
   9697],
  [0,
   49519,
   20,
```

```
   92,
   1288,
   16,
   10,
   18719,
   7,
   5,
   1073,
   43563,
   1419,
   656,
   42,
   76,
   19,
   5,
   13956,
   119]],
 'query': ["<s>Seppala's revenue increased by 0.2 % to EUR10.1 m.",
  "<s>At the request of Finnish media company Alma Media's newspapers, resear
ch manager Jari Kaivo",
  '<s>`` The new agreement is a continuation to theagreement signed earlier t
his year with the Lemm']}
```

My own data

In [18]:
```
# my_own_datasets = load_dataset("text", data_files={ "train": "path", "validat
```

In [19]:
```
# my_own_datasets
```

Now this for actual RLHF

In [20]:
```
def build_dataset(
        config,
        dataset_name="financial_phrasebank",
        input_min_text_length=2,
        input_max_text_length=8
):
    """
    Build dataset for training. This builds the dataset from `load_dataset`, or
    customize this function to train the model on its own dataset.

    Args:
        dataset_name (`str`):
            The name of the dataset to be loaded.

    Returns:
        dataloader (`torch.utils.data.DataLoader`):
            The dataloader for the dataset.
    """
    tokenizer           = AutoTokenizer.from_pretrained(config.model_name)
    tokenizer.pad_token = tokenizer.eos_token

    # load imdb with datasets

    #ds = load_dataset(dataset_name, split="train")
    ds  = load_dataset(dataset_name, 'sentences_50agree', split = "train")

    #ds = ds.rename_columns({"text": "sentence"})
    ds = ds.filter(lambda x: len(x["sentence"]) > 200, batched=False)

    input_size = LengthSampler(input_min_text_length, input_max_text_length)

    def tokenize(sample):
        sample["input_ids"] = tokenizer.encode( sample["sentence"]    )[: input
        sample["query"]     = tokenizer.decode( sample["input_ids"] )
        return sample

    ds = ds.map(tokenize, batched=False)
    ds.set_format(type="torch")
    return ds
```

In [21]:
```
dataset = build_dataset(config)
# dataset = load_dataset('financial_phrasebank', 'sentences_50agree')
```

In [22]:
```
def collator(data):
    return dict((key, [d[key] for d in data]) for key in data[0])
```

Load pre-trained GPT2 language models

We load the GPT2 model with a value head and the tokenizer. We load the model twice; the first model is optimized while the second model serves as a reference to calculate the KL-divergence from the starting point. This serves as an additional reward signal in the PPO training to make sure the optimized model does not deviate too much from the original language model.

In [23]:
```
model     = AutoModelForCausalLMWithValueHead.from_pretrained(config.model_name
ref_model = AutoModelForCausalLMWithValueHead.from_pretrained(config.model_name

tokenizer = AutoTokenizer.from_pretrained(config.model_name)

tokenizer.pad_token = tokenizer.eos_token
```

```
If you want to use `RobertaLMHeadModel` as a standalone, add `is_decoder=Tru
e.`
Some weights of RobertaForCausalLM were not initialized from the model checkp
oint at mrm8488/distilroberta-finetuned-financial-news-sentiment-analysis and
are newly initialized: ['lm_head.bias', 'lm_head.decoder.bias', 'lm_head.dens
e.bias', 'lm_head.dense.weight', 'lm_head.layer_norm.bias', 'lm_head.layer_no
rm.weight']
You should probably TRAIN this model on a down-stream task to be able to use
it for predictions and inference.
If you want to use `RobertaLMHeadModel` as a standalone, add `is_decoder=Tru
e.`
Some weights of RobertaForCausalLM were not initialized from the model checkp
oint at mrm8488/distilroberta-finetuned-financial-news-sentiment-analysis and
are newly initialized: ['lm_head.bias', 'lm_head.decoder.bias', 'lm_head.dens
e.bias', 'lm_head.dense.weight', 'lm_head.layer_norm.bias', 'lm_head.layer_no
rm.weight']
You should probably TRAIN this model on a down-stream task to be able to use
it for predictions and inference.
```

In [24]:
```
ppo_trainer = PPOTrainer(
                 config,
                 model,
                 ref_model,
                 tokenizer,
                 dataset=dataset,
                 data_collator=collator
)
```

```
Detected kernel version 3.10.0, which is below the recommended minimum of 5.
5.0; this can cause the process to hang. It is recommended to upgrade the ker
nel to the minimum version or higher.
```

Load BERT classifier (Reward Function)

We load a BERT classifier fine-tuned on the IMDB dataset.

In [25]:
```
torch.cuda.is_available()
```

Out[25]:
```
False
```

In [27]:
```
# torch.cuda.get_device_name()
```

In [28]:
```
device = ppo_trainer.accelerator.device
device
```

Out[28]:
```
device(type='cpu')
```

In [29]:
```
if ppo_trainer.accelerator.num_processes == 1:
    device = 0 if torch.cuda.is_available() else "cpu"  # to avoid a `pipeline

device
```

Out[29]:
```
'cpu'
```

In [30]:
```
ntiment_pipe = pipeline("sentiment-analysis", model="mrm8488/distilroberta-fine
```

In [31]:
```
text = "Operating profit totaled EUR 9.4 mn , down from EUR 11.7 mn in 2004 ."

sentiment_pipe(text, **sent_kwargs)
```

```
/home/chavan7/.local/lib/python3.8/site-packages/transformers/pipelines/text_
classification.py:104: UserWarning: `return_all_scores` is now deprecated,  i
f want a similar functionality use `top_k=None` instead of `return_all_scores
=True` or `top_k=1` instead of `return_all_scores=False`.
  warnings.warn(
```

Out[31]:
```
[[{'label': 'negative', 'score': 4.749904155731201},
  {'label': 'neutral', 'score': -2.4718210697174072},
  {'label': 'positive', 'score': -2.789074420928955}]]
```

In [32]:
```
text = "In the third quarter of 2010 , net sales increased by 5.2 % to EUR 205

sentiment_pipe(text, **sent_kwargs)
```

Out[32]:
```
[[{'label': 'negative', 'score': -2.473881244659424},
  {'label': 'neutral', 'score': -3.440216302871704},
  {'label': 'positive', 'score': 6.057607650756836}]]
```

In [33]:
```
text = "Pharmaceuticals group Orion Corp reported a fall in its third-quarter e

sentiment_pipe(text, **sent_kwargs)
```

Out[33]:
```
[[{'label': 'negative', 'score': 4.775065898895264},
  {'label': 'neutral', 'score': -2.3186452388763428},
  {'label': 'positive', 'score': -2.8229196071624756}]]
```

In [34]:
```
text = "When this investment is in place , Atria plans to expand into the Mosc

sentiment_pipe(text, **sent_kwargs)
```

Out[34]:
```
[[{'label': 'negative', 'score': -4.264527320861816},
  {'label': 'neutral', 'score': 4.1889967918396},
  {'label': 'positive', 'score': 0.7380496859550476}]]
```

Generation settings

For the response generation we just use sampling and make sure top-k and nucleus sampling are turned off as well as a minimal length.

In [35]:
```
gen_kwargs = {
         "min_length":   -1,
         "top_k":       0.0,
         "top_p":       1.0,
         "do_sample":  True,
         "pad_token_id": tokenizer.eos_token_id
}
```

Optimize model

Training loop

The training loop consists of the following main steps:

- Get the query and responses from the policy network (GPT-2)
- Get sentiments for query/responses from BERT
- Optimize policy with PPO using the (query, response, reward) triplet

In [36]:
```
output_min_length     = 4
output_max_length     = 16
output_length_sampler = LengthSampler(output_min_length, output_max_length)
```

In [37]:
```
generation_kwargs = {
    "min_length":     -1,
    "top_k":          0.0,
    "top_p":          1.0,
    "do_sample":    True,
    "pad_token_id": tokenizer.eos_token_id,
}
```

In [38]:
```
## ppo_trainer.config.steps = 100    ## 20,000
ppo_trainer.config.steps
```

Out[38]:
```
20000
```

In [39]:
```
for epoch, batch in tqdm(enumerate(ppo_trainer.dataloader)):
    query_tensors = batch["input_ids"]

    print(query_tensors)
    print(len(query_tensors))
    if epoch == 1:
        break
```

```
1it [00:00, 11.19it/s]
```

```
[tensor([    0,   597,  5246,  1173, 24753, 18419]), tensor([   0,  133, 936
8,  128,   29, 8610]), tensor([    0, 14836]), tensor([   0,  597, 5246]), te
nsor([   0, 1640, 4516,  510,  491, 4839,  111]), tensor([  0, 597]), tensor
([    0, 17986]), tensor([  0, 133]), tensor([   0,  133, 3457, 2156, 2964]),
tensor([   0,  597, 5246, 1173, 3296]), tensor([   0,  250, 4779, 2186]), ten
sor([    0, 14229,     5,   375,  2202,    24,    34]), tensor([    0,   725,
16416, 23617,   100]), tensor([    0, 15685,   128,    29,   299, 11255,  673
8]), tensor([   0, 1640, 4516,  510,  491, 4839]), tensor([    0, 41858,   13
8,  4832,   255,  5810]), tensor([   0, 5320, 4330, 2271, 2527]), tensor([
0, 133]), tensor([   0, 2606,  510, 2926]), tensor([    0, 24476,   168,  723
6]), tensor([    0,   133,  4069,     9,    70,   518, 10298]), tensor([
0,   133,   786,    12, 13139]), tensor([    0,   597,  5246,  1173, 11767,
2643,  6596]), tensor([    0, 41024, 18366,   468]), tensor([    0, 31161]),
tensor([   0,  597, 5246, 1173]), tensor([   0,  597, 5246]), tensor([    0,
36735,  4832,  2156]), tensor([  0, 597]), tensor([   0, 3908,    5,   92,  9
31, 2195]), tensor([    0,   725,   791, 14469,  2620]), tensor([    0,   13
3, 21533,   168,   585]), tensor([    0,  6433,  3850, 10370, 42204]), tensor
([    0, 11350, 10612,  5969, 39261,    12]), tensor([   0, 1225,  830]), ten
sor([    0,   597,  5246,  1173, 32902]), tensor([  0, 133]), tensor([    0,
25546]), tensor([    0,   597,  5246,  1173,  7746,   462, 10544]), tensor([
0,  597, 5246, 1173,  613]), tensor([   0,  243,  554,   19, 2257]), tensor([
0,  565, 5810]), tensor([    0,   133, 38507]), tensor([    0, 47380,  1161,
647,    31,     5, 12168]), tensor([  0, 996]), tensor([   0, 1640]), tensor
([  0, 597]), tensor([    0, 15791,  6407,    34,   411]), tensor([   0, 248
1,  644]), tensor([    0,   133,  5448, 14258]), tensor([    0, 23565,   783,
16,   546,     7,  3754]), tensor([   0,  597, 5246, 1173, 5516, 1743]), tens
or([   0,  487, 3109,  636]), tensor([   0,  133, 1355,   67, 1171, 3931]), t
ensor([  0, 133]), tensor([    0, 12271,  4358,     8,  6955]), tensor([
0,   530,  1439, 18649, 19938]), tensor([    0, 14693]), tensor([   0,  597,
5246, 1173,  827]), tensor([  0, 134]), tensor([   0,  133, 3566]), tensor([
0,   133, 15785,  4071,    33,  3978]), tensor([    0, 37729,   271,  7223,
2156,   854,  2068]), tensor([    0, 11350, 10612]), tensor([   0,  347, 538
4, 1242, 3204]), tensor([   0,  133, 2676]), tensor([    0, 44002,    11,  35
03,    21]), tensor([   0,  487, 1988, 3019]), tensor([    0, 10169,  4235,
330,  3852,   817]), tensor([    0,   133,   138,    26,  4995, 12460]), tens
or([    0, 33939,  3855,  1797]), tensor([    0, 20420,  1295,  1963]), tenso
r([    0, 29991,  4458,  2757]), tensor([   0,  133, 2319,  923,    9]), tens
or([    0,  2744, 29254]), tensor([   0, 1640, 4516]), tensor([    0, 49519,
20, 31417]), tensor([   0, 1121,   10,  485, 1194,   19,    5]), tensor([  0,
717]), tensor([    0,   565,  1250,  4291,  8897,  1910, 16763]), tensor([
0, 20689, 18121]), tensor([    0, 24789,   545,   550]), tensor([    0,   48
7, 13967,  4541,    26,    24,    40]), tensor([    0,  1121, 10906, 15739, 1
2047,  1820]), tensor([    0,   306,   902,  1466,   111, 21533]), tensor([
0, 16837,  1839,    67,  4865,   400]), tensor([    0, 46717]), tensor([
0, 49519,   166,    40,   535,     7]), tensor([  0, 530]), tensor([    0, 22
862,  2644]), tensor([    0, 22086,     7]), tensor([    0, 44084]), tensor([
0, 28873,  1902,     9,   800,  7000]), tensor([   0,  133, 6994,   26,   6
3]), tensor([  0, 250]), tensor([   0, 1640, 4516]), tensor([   0, 4993, 568
7,    9,    5, 3857]), tensor([    0,   133, 21533,   138,  1088,    63]), te
nsor([  0, 698, 902]), tensor([    0, 33477,   545,   494,  1824]), tensor([
0, 43787,   268,    26,     5,   458]), tensor([   0, 4444,  846]), tensor([
0,   133, 20785, 40556,  2835,  1523, 13154]), tensor([   0, 2606,  510]), te
nsor([    0, 10494,   128]), tensor([    0, 14559]), tensor([    0, 10653,  1
999,  3037]), tensor([   0,  597, 5246]), tensor([   0, 2118, 6725, 5945,  50
0, 2889]), tensor([   0, 1620,    5,  232,  917,   11]), tensor([   0, 7083,
1916, 2454]), tensor([    0,  1121, 10906,  2824,  3412,  3080, 15781]), tens
or([    0, 49519,    20,  2175,  4366, 22715,    31]), tensor([    0, 48767,
```

```
102,    34,   931]), tensor([    0, 36417, 13967,   128]), tensor([   0, 112
1]), tensor([    0, 11108, 21676,  2175]), tensor([   0, 1640]), tensor([
0, 19877, 42292,   384]), tensor([    0,   133, 29147,   104, 30445]), tensor
([    0, 14693,     7, 21533]), tensor([    0, 47559,   680,  7653]), tensor
([   0,  574, 4989,  362, 1998]), tensor([    0, 22886,  1641,   354, 46707,
16613,  3593]), tensor([    0, 22086,     7]), tensor([  0, 597]), tensor([
0, 4148,    5, 1453,    9]), tensor([    0, 20839,     5])]
128
[tensor([   0,  597, 5246, 1173]), tensor([  0, 133, 265,  34, 647,   9]), te
nsor([   0, 2895]), tensor([   0,  597, 5246, 1173]), tensor([    0,   133, 2
2873,    11,    63]), tensor([    0, 14693]), tensor([   0,  597, 5246]), ten
sor([    0, 14465,   975, 16802,   221]), tensor([    0,   565,  5810,   139,
11, 27749,    16]), tensor([   0,  133, 2443,  606,   80]), tensor([    0,
605,  1344,  8757,  7606, 10502,   883]), tensor([   0,  597, 9968,   11]), t
ensor([   0,  134,  587, 1466,  111]), tensor([   0, 1922]), tensor([    0, 1
1321, 17692,    11,  6193]), tensor([   0,  597, 5246, 1173]), tensor([  0, 1
33]), tensor([    0, 23031, 22714,  1001,  2543,  3291]), tensor([    0, 1469
3,     7,     5,   138]), tensor([    0,   133,   418,    40,    28,  1240, 4
5518]), tensor([   0,  597, 5246]), tensor([    0, 14696,  3807,  2331,   12
8,    29]), tensor([   0,  448, 2580,  102]), tensor([   0, 1244]), tensor([
0,  597, 5246, 1173, 1663]), tensor([    0, 31004,   811,   786,    12,  769
9]), tensor([    0, 34311]), tensor([    0, 38741,   467,  9509, 12169]), ten
sor([    0,   487, 11200,  2371, 29915]), tensor([  0, 282]), tensor([    0,
27814,  3760,  3206]), tensor([  0, 398]), tensor([    0,   541,   759,  233
8,   111, 21533]), tensor([    0,  3972,  3042, 32992,  3247,  1330,     7]),
tensor([   0, 2481,  644, 1466,  111]), tensor([    0, 12905, 10318]), tensor
([    0,   725,   791, 14469,  2620,  7140,   100]), tensor([   0,  597, 524
6, 1173, 3689, 4403,  248]), tensor([    0, 17986,   462, 10544]), tensor([
0, 49519, 12786,   242,  4541]), tensor([    0, 15721,   647,     9, 21533,
764]), tensor([  0, 133]), tensor([    0, 28152]), tensor([  0, 597]), tensor
([    0, 10285,   186,  2156,     5]), tensor([    0,    29, 14680,  3223, 23
765]), tensor([   0, 1360,  494, 1466,  111, 6765, 8922]), tensor([    0,   3
47, 12019,     7,     5,  5405]), tensor([    0, 19923,  2963]), tensor([
0,  133, 1288,   21]), tensor([    0,   597,  5246,  1173, 28104]), tensor([
0, 14693,     7, 21533,  1911]), tensor([  0, 133]), tensor([   0,  597, 524
6, 1173,  476]), tensor([    0, 49519]), tensor([    0, 40529, 42310,  2156,
312,     4]), tensor([   0,  133, 5195,   34, 2740]), tensor([    0, 49519,
4028,    76,  2156,  1081]), tensor([    0,   597,  5246,  1173, 11451,   41
4,  4358]), tensor([  0, 597]), tensor([   0,  401,  779, 1824]), tensor([
0, 34027,  1245,   128,    29]), tensor([   0,  975, 2068]), tensor([   0,  1
33, 3689, 5406]), tensor([  0, 246, 902]), tensor([    0,  4741, 21953,  150
3,    16,   145]), tensor([   0,  347, 4040,  139,  128,   29]), tensor([
0, 34027,  1245,    12]), tensor([    0, 14693,     7,    41]), tensor([   0,
406,  494, 1466,  111]), tensor([    0, 46769,  5516,   785,   680]), tensor
([   0,  250, 4779, 2186,    7, 1702, 5374]), tensor([    0,  4794,   104,  6
282, 17088,    34,    57]), tensor([   0,  133, 2087,    9,    5,  695]), ten
sor([  0, 597]), tensor([    0, 29891,  3293,  3586,   384]), tensor([  0, 13
3, 568,   7]), tensor([   0, 2606,  510]), tensor([   0,  179, 1209,  134, 18
24,  504,  392]), tensor([  0, 133, 381]), tensor([    0,   597,  5246,  117
3, 32053,     8,  2225]), tensor([   0,  597, 5246, 1173, 2859,   12, 4903]),
tensor([    0, 35469]), tensor([    0, 15791]), tensor([   0,  134,  181,
4,  119,    4, 1505]), tensor([   0,  347, 5174,   21, 2885,   15]), tensor([
0, 25869,  2527, 17311,   267]), tensor([   0, 1640, 4516]), tensor([    0,
1121, 27815,    19,     5]), tensor([   0,  133, 9367,   16]), tensor([    0,
133, 23253,    12,   805]), tensor([   0, 2481,  779]), tensor([    0,   597,
5246,  1173,  3296, 18121]), tensor([    0, 10105, 14300,   531,    28, 1426
7]), tensor([    0, 46170, 13548, 29915, 28275,  7857]), tensor([    0, 1724
```

```
5,     5,  1288, 40109]), tensor([    0, 17245,     5,  1110,     9,     5,
160]), tensor([    0, 18348,  8063]), tensor([   0,  597, 5246, 1173,  184]),
tensor([  0, 844, 779]), tensor([    0,   487, 13967,  4541,  1913,     4,
34]), tensor([   0, 4148,    5]), tensor([    0, 20930,    15,     5,    7
8]), tensor([   0,  133,  138, 4542,  916]), tensor([   0,  597, 5246, 117
3]), tensor([   0, 1244,  759]), tensor([    0, 20981, 42987, 44581, 17311,
267,   579]), tensor([    0,  2336, 14465,  3037,   438,  4062, 13181]), tens
or([    0,   133, 36007,  2156,  4241,    19,     5]), tensor([    0,   597,
5246,  1173,  3942, 24753, 18419]), tensor([    0,  1620, 36692]), tensor([
0,   597,  5246,  1173,  5195, 14533,  2456]), tensor([    0,   347,   868, 1
4675,  5778]), tensor([  0, 133]), tensor([    0,  2890,   772,  2156,  1824,
21533]), tensor([    0, 45356,    29]), tensor([    0,   597,  5246,  1173, 2
8630]), tensor([   0,  597, 5246, 1173]), tensor([   0,  597, 5246]), tensor
([    0,   844,   779,  1824,   111, 21533]), tensor([   0,  133,  758, 5175,
539,  138]), tensor([    0,   133,  5195,  2319,    14,     5, 16064]), tenso
r([    0,   597,  5246,  1173, 28630,   885,  2001]), tensor([   0,  597, 524
6]), tensor([   0,  133, 4103]), tensor([  0, 597]), tensor([    0,   133,
458,   696,  2156, 47158]), tensor([    0, 12582, 11192,  3037])]
128
```

In [40]:
```
for epoch, batch in tqdm(enumerate(ppo_trainer.dataloader)):
    query_tensors = batch["input_ids"]
    print(epoch)
    print(batch)
    print('**********************')
    print('**********************')
    print('**********************')
    print('**********************')
    #### Get response from gpt2
    response_tensors = []
    for query in query_tensors:
        gen_len                                = output_length_sampler()
        generation_kwargs["max_new_tokens"] = gen_len
        response                               = ppo_trainer.generate(query, **ge
        response_tensors.append( response.squeeze()[-gen_len:] )
    batch["response"] = [ tokenizer.decode(r.squeeze()) for r in response_tens
    print(batch)
    if epoch == 1:
        break
```

```
nowhere widen yet', 'ilionß outswk DET corridors interchangeable lane', ' l
atitudeamongripprev charm eternal nowhere anonymity insider anonymity Orbit
al', ' insider nowhere insider eternity geographically unchrunners', ' unca
nny bes geographically lane lane interchangeable interchangeableabilitiesCo
lor discretion nowhereñactions lane latitude', 'abbling eloqugemspell nonex
insider', ' runway Deepexc asymmEEP Studios narrowedurai Arrayabilities Gov
ernors', ' repszeltain bona Swap circum', ' engulf INS elbows Animated', '}
\\ cushion retake Alive', 'orem Governors exoner infinitely openerometry ea
ves obscure', ' shortcuts differentiate lane Seg lane midway insider anonym
ity interchangeable', ' uniquelyips elbows wherever interchangeable interch
angeableavour lane lane', ' cushion inappropriatelyuddin episodeoks GMplotw
k Ü unreasonable', ' interchangeable gulf interchangeable lane Somewhere in
sider interchangeable insider hereafter geographically setup unch', ' Edito
r Appropri Previous Rout Round uncanny shoe Prev', ' OpenGL arr×illing gro
p', ' lane Gadget vault counsel uncanny insider lane interchangeable', ' to
rque obstacle confid Deep interchangeable broaden broadenindingREM nowhere
anonymity anonymity uncanny', ' fashioned latitude geographicallyhips intri
g tint Channel latitude Deep× grasping latitude engulf barric', ' uniquely
infinite allotted gropchel lane lane nowhere Swap latitude Parkway Shut Par
kway eternity', ' enigmatic shoulders insider Lans insider spotlight Places
```

In [41]:
```
batch.keys()
```

Out[41]:
```
dict_keys(['label', 'input_ids', 'query', 'response'])
```

Compute sentiment score

In [42]:
```
batch["query"]
```

```
 '<s>`` We have',
 '<s>Karachi, Sept',
 '<s>Finnish consumer',
 '<s>Incap Corporation Stock Exchange Release',
 '<s>Risk exposure by Non-',
 '<s>Under the agreement Benef',
 '<s>According to',
 '<s>NAV',
 '<s>``',
 '<s>Music is provided',
 '<s>In the Czech',
 '<s>KESKO CORPORATION',
 '<s>Marimekko makes',
 '<s>YIT',
 '<s>The international electronic industry company',
 '<s>In',
 '<s>Mr. Atul Chopra',
 '<s>SINGAP',
 '<s>HUHTAMAKI',
 '<s>The Finnish government announced'
```

In [43]:
```
batch["response"]
```

```
 ' shoulder insider Governors Recall Known dearly oval Parkwaymill foresee
fielder Parkway anticipating unch midway',
 'oks Vij opener Governors midwayDJ',
 ' gelTERN nowhere labyrinth unch Parkway Albany',
 ' Shut Compl Cosmetic SPECIAL',
 ' conjecture conjecture Gim mirrors',
 ' Oval lane flaw maj ProcessporEEPaps insider geographically',
 ' scramble agon automakersdrive',
 ' interchangeable infinite AppEEP shortcuts OvalBUS recesslights underrate
d',
 ' insider interchangeable interchangeable unimagin Editor wrencheely',
 ' conjecture geographically bridges doomrim ways midway nowhere nowhere no
where Somewhere lane Parkway circular',
 ' lane bridges insider bisexual geographically laneridor',
 ' rim infinite nowhere upfront nowhereoksopers interchangeable interchange
able frequent insider Recall',
 ' lane insider Recall mirror limitless flirt Parkway Parkway flawificantly
interchange Parkway',
 ' interchangeable Array bes lane Wayocal insider engulf alphabet nowhere G
rip elbow',
```

In [44]:
```
texts = [ q + r for q, r in zip(batch["query"], batch["response"]) ]
```

In [45]:
```
texts
```

```
geographically ',
 '<s>Incap Corporation Stock Exchange Release rearr maneuvermeg brazenrupul
ous cunningoks',
 '<s>Risk exposure by Non-wk MUCH gestation Parkway antibiotic Included uri
nary coy',
 '<s>Under the agreement Benefoksennes infinite elevation barrier',
 '<s>According to enorm scenicpit intric insider anonymity',
 '<s>NAV Mov engulfArtist latituderup× conce runway engulf Slightly Parkway
Oval uncanny spiritually nowhere',
 '<s>``ß runway runway Somewhere curvllular Mur eaves widening epoch Alleg
widen Alleg',
 '<s>Music is provided grop categor shadowoks setup extended gulf precedEEP
flaw Tripept.), latitude',
 '<s>In the Czech infinite fielderrupulous lane Somewhere',
 '<s>KESKO CORPORATION presses grop Sneak Sneak spared Halhun Hal',
 "<s>Marimekko makes bona citation jumper:'function obfusc Swap insider Som
ethingITCH autop insider KEY Become",
 '<s>YIT lane continents interchangeable Oval interchangeable clue intercha
ngeablemoving × Alvin',
 '<s>The international electronic industry company geographically oval Swap
```

In [46]:
```
pipe_outputs = sentiment_pipe(texts, **sent_kwargs)
pipe_outputs
```

```
  {'label': 'positive', 'score': -3.199592113494873}],
 [{'label': 'negative', 'score': -2.527575969696045},
  {'label': 'neutral', 'score': 6.520326614379883},
  {'label': 'positive', 'score': -3.3524115085601807}],
 [{'label': 'negative', 'score': -2.9812164306640625},
  {'label': 'neutral', 'score': 6.656062126159668},
  {'label': 'positive', 'score': -2.9706711769104004}],
 [{'label': 'negative', 'score': -2.6016390323638916},
  {'label': 'neutral', 'score': 6.543796539306641},
  {'label': 'positive', 'score': -3.280642509460449}],
 [{'label': 'negative', 'score': -2.526651620864868},
  {'label': 'neutral', 'score': 5.838469505310059},
  {'label': 'positive', 'score': -2.859650135040283}],
 [{'label': 'negative', 'score': -2.879009485244751},
  {'label': 'neutral', 'score': 6.04233455657959},
  {'label': 'positive', 'score': -2.603118419647217}],
 [{'label': 'negative', 'score': -2.8631174564361572},
  {'label': 'neutral', 'score': 6.699374675750732},
  {'label': 'positive', 'score': -3.1245744228363037}],
 [{'label': 'negative', 'score': -2.9103055000305176},
```

In [47]:
```
rewards = [ torch.tensor(output[1]["score"]) for output in pipe_outputs]
rewards
```

```
 tensor(6.1578),
 tensor(6.6126),
 tensor(6.4347),
 tensor(6.5861),
 tensor(6.1342),
 tensor(6.6648),
 tensor(4.3954),
 tensor(6.2609),
 tensor(6.5741),
 tensor(6.3860),
 tensor(5.9279),
 tensor(6.3543),
 tensor(6.4896),
 tensor(6.3386),
 tensor(6.6811),
 tensor(6.4053),
 tensor(1.0354),
 tensor(5.8819),
 tensor(6.5564)]
```

In [48]:
```
len(rewards)
```

Out[48]:
```
128
```

In [ ]:
```

```

In [49]:
```
for epoch, batch in tqdm(enumerate(ppo_trainer.dataloader)):
    query_tensors = batch["input_ids"]
    print(epoch)

    #### Get response from gpt2
    response_tensors = []
    for query in query_tensors:
        gen_len                                = output_length_sampler()
        generation_kwargs["max_new_tokens"] = gen_len
        response                               = ppo_trainer.generate(query, **ge
        response_tensors.append( response.squeeze()[-gen_len:] )
    batch["response"] = [ tokenizer.decode(r.squeeze()) for r in response_tens

    #### Compute sentiment score
    texts = [q + r for q, r in zip(batch["query"], batch["response"])]
    pipe_outputs = sentiment_pipe(texts, **sent_kwargs)
    rewards = [ torch.tensor(output[1]["score"]) for output in pipe_outputs]

    #### Run PPO step
    stats = ppo_trainer.step(
                    query_tensors,
                    response_tensors,
                    rewards
    )
    ppo_trainer.log_stats(stats, batch, rewards)
```

```
0it [00:00, ?it/s]

0

/home/chavan7/.local/lib/python3.8/site-packages/trl/trainer/ppo_trainer.py:1
212: UserWarning: The average ratio of batch (93.66) exceeds threshold 10.00.
Skipping batch.
  warnings.warn(
/home/chavan7/.local/lib/python3.8/site-packages/trl/trainer/ppo_trainer.py:1
212: UserWarning: The average ratio of batch (367.75) exceeds threshold 10.0
0. Skipping batch.
  warnings.warn(
1it [04:36, 276.00s/it]

1

2it [09:11, 275.56s/it]

2

3it [13:50, 277.19s/it]

3

4it [19:17, 289.39s/it]
```

In [51]:
```
# torch.cuda.get_device_name(0)
```

One can observe how the model starts to generate more positive outputs after a few optimisation steps.

Note: Investigating the KL-divergence will probably show that at this point the model has not converged to the target KL-divergence, yet. To get there would require longer training or starting with a higher initial coefficient.

Let's inspect some examples from the IMDB dataset. We can use model_ref to compare the tuned model model against the model before optimisation

In [52]:
```
#### get a batch from the dataset
bs                  = 16
game_data           = dict()
```

In [53]:
```
game_data
```

Out[53]:
```
{}
```

In [54]:
```
dataset.set_format("pandas")
```

In [55]:
```
df_batch          = dataset[:].sample(bs)
df_batch
```

Out[55]:

	sentence	label	input_ids	query
602	Finnish Scanfil , a systems supplier and contr...	0	[0, 597, 5246, 1173]	<s>Finnish
291	`` In the newly formed company YIT Stavo the l...	2	[0, 49519]	<s>``
146	NORDIC BUSINESS REPORT-26 June 2006-Metso Corp...	2	[0, 487, 11200, 2371, 29915]	<s>NORDIC BUS
311	in Q1 2010 18 May 2010 - Finnish electrical co...	2	[0, 179, 1209, 134, 1824, 504, 392]	<s>in Q1 2010 18 May
294	in Q1 '10 19 April 2010 - Finnish forest machi...	2	[0, 179]	<s>in
173	The Daily Graphic newspaper , in October , rep...	1	[0, 133]	<s>The
340	The carrier said its Area travel agency is to ...	1	[0, 133, 6994, 26, 63]	<s>The carrier said its
205	The Lemminkainen Group , headquartered in Hels...	1	[0, 133, 13956, 119, 4291, 1851, 225]	<s>The Lemminkainen
535	25 March 2011 - Finnish electronics contract m...	0	[0, 1244]	<s>25
470	treatment products in Usa , Canada , Mexico , ...	1	[0, 37558, 785, 11]	<s>treatment products in
224	The Annual General Meeting approved that the y...	1	[0, 133, 7453]	<s>The Annual
560	20 October 2010 - Finnish environmental manage...	0	[0, 844, 779]	<s>20 October
330	Finnish broadband data communication systems p...	2	[0, 597, 5246, 1173, 11451, 414, 4358]	<s>Finnish broadband data communication
496	LCS 's services cover the whole life cycle of ...	1	[0, 34311]	<s>LCS
595	Since the association 's data do not cover sal...	0	[0, 11321, 5, 5259, 128, 29, 414]	<s>Since the association's data
21	TELECOMWORLDWIRE-7 April 2006- TJ Group Plc sel...	1	[0, 6433, 3850, 10370, 42204, 18048]	<s>TELECOMWORLD

In [56]:
```
game_data["query"] = df_batch["query"].tolist()
query_tensors      = df_batch["input_ids"].tolist()
```

In [57]:
```
response_tensors_ref, response_tensors = [], []
```

In [58]:
```
#### get response from gpt2 and gpt2_ref
for i in range(bs):
    gen_len = output_length_sampler()

    output  = ref_model.generate(
        torch.tensor(query_tensors[i]).unsqueeze(dim=0).to(device), max_new_tol
    ).squeeze()[-gen_len:]
    response_tensors_ref.append(output)


    output = model.generate(
        torch.tensor(query_tensors[i]).unsqueeze(dim=0).to(device), max_new_tol
    ).squeeze()[-gen_len:]
    response_tensors.append(output)
```

In [59]:
```
#### decode responses
game_data["response (before)"] = [tokenizer.decode(response_tensors_ref[i]) fo
game_data["response (after)"]  = [tokenizer.decode(response_tensors[i]) for i
```

In [60]:
```
#### sentiment analysis of query/response pairs before/after
texts = [q + r for q, r in zip(game_data["query"], game_data["response (before)
game_data["rewards (before)"] = [output[1]["score"] for output in sentiment_pi
```

```
/home/chavan7/.local/lib/python3.8/site-packages/transformers/pipelines/text_
classification.py:104: UserWarning: `return_all_scores` is now deprecated,  i
f want a similar functionality use `top_k=None` instead of `return_all_scores
=True` or `top_k=1` instead of `return_all_scores=False`.
  warnings.warn(
```

In [61]:
```
texts = [q + r for q, r in zip(game_data["query"], game_data["response (after)"
game_data["rewards (after)"] = [output[1]["score"] for output in sentiment_pip
```

In [62]:
```
# store results in a dataframe
df_results = pd.DataFrame(game_data)
df_results
```

Out[62]:

	query	response (before)	response (after)	rewards (before)	rewards (after)
0	<s>Finnish	Klu listener "(ringsedom)--ctuary Flags Chan...	anonymity quiz Stretch Array Series Sneak all...	6.653607	6.368655
1	<s>``	berus whistleblowerorama SoundsglersESTesides ...	ß kick Imag Lynchbentchart bes shortcuts portr...	6.483539	6.392416
2	<s>NORDIC BUS	PenguinscookBulsets	latitude filibuster widened Sag	6.537941	0.484987
3	<s>in Q1 2010 18 May	headlights Fav RockiesINCctuary thriveSnapadd...	drastically SneakEEPrupulous SPECIAL outsrupu...	6.526322	6.022478
4	<s>in	sets Chancellor Riders ear Based listener wint...	latitude many cumbersome Locked subter consta...	6.599554	3.030718
5	<s>The	Sanctuary Klu rewards Standingrivalquerspring...	layout recall KEY latitude latitudeterior unc...	6.419761	6.733101
6	<s>The carrier said its	Chancellor mayorsills BEL brand Klu Frames Ba...	latitudeooth brillianceOWSrenchrupulous arden...	6.556527	4.173765
7	<s>The Lemminkainen	attachments winters Klu potatoesiking Throne ...	indisc AL identifier Extended uncanny latitud...	6.569796	6.566141
8	<s>25	paraphsets Reign Flags Klu THANK plunge Avg	nowhere Ital obscure uncanny Agingrupulous AP...	6.418313	6.269688
9	<s>treatment products in	KB Bucks Helmet poses Crimeasets Prosecutors ...	dogged plottedDriveabbling encl Garden infini...	6.425732	6.105270
10	<s>The Annual	COL Klutesy boil begg	geographically anonymity tailor latitude lati...	6.729272	6.705943
11	<s>20 October	Flagseland PANriers Klu 09 winters Crimean Ri...	Tough advertisers Suns gel interchangeableuou...	6.617059	6.654399
12	<s>Finnish broadband data communication	Klu whine Iro Sanctuarysetssets BELsets Riders	osure plains infinitely autonomy latitude sche...	6.560050	6.383207
13	<s>LCS	CrimeanALKListener benefit tresp Names rings ...	ImagColor interchangeable×ikan unch Stretch b...	5.170542	5.311996
14	<s>Since the association's data	ggingcca attaches Yuanstrings Ruleedom FlagsmA...	Silicon PrevChip Stretch automakers geographi...	6.649787	6.609080
15	<s>TELECOMWORLD	Klu bends rings Reign kettle Kroncesaundering...	eternity latitude latitude Deep latitude lati...	6.224792	6.652760

In [63]:
```
print("mean:")
display(df_results[["rewards (before)", "rewards (after)"]].mean())
print()
print("median:")
display(df_results[["rewards (before)", "rewards (after)"]].median())
```

```
mean:

rewards (before)    6.446412
rewards (after)     5.654038
dtype: float64


median:

rewards (before)    6.547234
rewards (after)     6.375931
dtype: float64
```

In [64]:
```
## model.save_pretrained(    "gpt2-imdb-pos-v2", push_to_hub=True)
## tokenizer.save_pretrained("gpt2-imdb-pos-v2", push_to_hub=True)

model.save_pretrained(    "gpt2-imdb-pos-v2", push_to_hub=False)
tokenizer.save_pretrained("gpt2-imdb-pos-v2", push_to_hub=False)
```

Out[64]:
```
('gpt2-imdb-pos-v2/tokenizer_config.json',
 'gpt2-imdb-pos-v2/special_tokens_map.json',
 'gpt2-imdb-pos-v2/vocab.json',
 'gpt2-imdb-pos-v2/merges.txt',
 'gpt2-imdb-pos-v2/added_tokens.json',
 'gpt2-imdb-pos-v2/tokenizer.json')
```

In [ ]:

In [ ]:

In [ ]: