Assignment 3

Install the Transformers, Datasets, and Evaluate libraries to run this notebook.

```
!pip install datasets evaluate transformers[sentencepiece]
!apt install git-lfs
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting datasets
  Downloading datasets-2.8.0-py3-none-any.whl (452 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 452.9/452.9 KB 16.9 MB/s eta 0:00:00
Collecting evaluate
  Downloading evaluate-0.4.0-py3-none-any.whl (81 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 81.4/81.4 KB 11.3 MB/s eta 0:00:00
Collecting transformers[sentencepiece]
  Downloading transformers-4.25.1-py3-none-any.whl (5.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.8/5.8 MB 63.2 MB/s eta 0:00:00
Requirement already satisfied: pyarrow>=6.0.0 in /usr/local/lib/python3.8/dist-packages (from datasets) (9.0.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.8/dist-packages (from datasets) (6.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.8/dist-packages (from datasets) (1.21.6)
Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python3.8/dist-packages (from datasets) (2.25.1)
Collecting multiprocess
  Downloading multiprocess-0.70.14-py38-none-any.whl (132 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 132.0/132.0 KB 15.2 MB/s eta 0:00:00
Requirement already satisfied: dill<0.3.7 in /usr/local/lib/python3.8/dist-packages (from datasets) (0.3.6)
Requirement already satisfied: packaging in /usr/local/lib/python3.8/dist-packages (from datasets) (21.3)
Requirement already satisfied: tqdm>=4.62.1 in /usr/local/lib/python3.8/dist-packages (from datasets) (4.64.1)
Requirement already satisfied: pandas in /usr/local/lib/python3.8/dist-packages (from datasets) (1.3.5)
Collecting xxhash
  Downloading xxhash-3.2.0-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (213 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 213.0/213.0 KB 19.8 MB/s eta 0:00:00
Collecting responses<0.19
  Downloading responses-0.18.0-py3-none-any.whl (38 kB)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.8/dist-packages (from datasets) (3.8.3)
Collecting huggingface-hub<1.0.0,>=0.2.0
  Downloading huggingface_hub-0.11.1-py3-none-any.whl (182 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 182.4/182.4 KB 15.5 MB/s eta 0:00:00
Requirement already satisfied: fsspec[http]>=2021.11.1 in /usr/local/lib/python3.8/dist-packages (from datasets) (2022.11.0)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.8/dist-packages (from transformers[sentencepiece]) (2022.6
Requirement already satisfied: filelock in /usr/local/lib/python3.8/dist-packages (from transformers[sentencepiece]) (3.8.2)
Collecting tokenizers!=0.11.3,<0.14,>=0.11.1
  Downloading tokenizers-0.13.2-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (7.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.6/7.6 MB 66.2 MB/s eta 0:00:00
Requirement already satisfied: protobuf<=3.20.2 in /usr/local/lib/python3.8/dist-packages (from transformers[sentencepiece]) (3.19.6)
Collecting sentencepiece!=0.1.92,>=0.1.91
  Downloading sentencepiece-0.1.97-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 73.4 MB/s eta 0:00:00
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (22.2.0)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (1.8.2)
Requirement already satisfied: charset-normalizer<3.0,>=2.0 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (2.1.1
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (1.3.3)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (6.0.3)
Requirement already satisfied: async-timeout<5.0,>=4.0.0a3 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (4.0.2)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (1.3.1)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.8/dist-packages (from huggingface-hub<1.0.0,>=0.2
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.8/dist-packages (from packaging->datasets) (3.0.9)
Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.8/dist-packages (from requests>=2.19.0->datasets) (4.0.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/dist-packages (from requests>=2.19.0->datasets) (2022.12
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packages (from requests>=2.19.0->datasets) (2.10)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.8/dist-packages (from requests>=2.19.0->datasets) (1.24
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.13-py2.py3-none-any.whl (140 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 140.6/140.6 KB 17.4 MB/s eta 0:00:00
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.8/dist-packages (from pandas->datasets) (2022.7)
```

```
from datasets import load_dataset, DatasetDict

ds_train = load_dataset("huggingface-course/codeparrot-ds-train", split="train")
ds_valid = load_dataset("huggingface-course/codeparrot-ds-valid", split="validation")

raw_datasets = DatasetDict(
    {
        "train": ds_train.shuffle(seed=42).select(range(50000)),
        "valid": ds_valid.shuffle(seed=42).select(range(500))
    }
)
```

```
raw_datasets
```

```
WARNING:datasets.builder:Using custom data configuration huggingface-course--codep
Downloading and preparing dataset json/huggingface-course--codeparrot-ds-train to
```

```
Downloading data files:                                    1/1 [02:48<00:00,
100%                                                       168.87s/it]
Downloading data:                                  8.25G/8.25G [02:48<00:00,
100%                                                       69.7MB/s]
```

```
WARNING:datasets.download.download_manager:Computing checksums of downloaded files
```

```
Computing checksums:                                       1/1 [00:45<00:00,
100%                                                       45.70s/it]
Extracting data files: 100%                                1/1 [00:00<00:00, 38.14it/s]
```

```
Dataset json downloaded and prepared to /root/.cache/huggingface/datasets/huggingf
WARNING:datasets.builder:Using custom data configuration huggingface-course--codep
Downloading and preparing dataset json/huggingface-course--codeparrot-ds-valid to
```

```
Downloading data files:                                    1/1 [00:04<00:00,
100%                                                       4.21s/it]
Downloading data:                                  46.1M/46.1M [00:01<00:00,
100%                                                       41.5MB/s]
Extracting data files: 100%                                1/1 [00:00<00:00, 51.73it/s]
```

```
Dataset json downloaded and prepared to /root/.cache/huggingface/datasets/huggingf
DatasetDict({
```

```
for key in raw_datasets["train"][0]:
    print(f"{key.upper()}: {raw_datasets['train'][0][key][:200]}")
```

```
REPO_NAME: ThomasMiconi/htmresearch
PATH: projects/feedback/feedback_sequences.py
COPIES: 2
SIZE: 26875
CONTENT:
# Numenta Platform for Intelligent Computing (NuPIC)
# Copyright (C) 2016, Numenta, Inc.  Unless you have an agreement
# with Numenta, Inc., for a separate license for this software code, the
# follo
LICENSE: agpl-3.0
```

```
from transformers import AutoTokenizer

context_length = 128
tokenizer = AutoTokenizer.from_pretrained("huggingface-course/code-search-net-tokenizer")

outputs = tokenizer(
    raw_datasets["train"][:2]["content"],
    truncation=True,
    max_length=context_length,
    return_overflowing_tokens=True,
    return_length=True,
)

print(f"Input IDs length: {len(outputs['input_ids'])}")
print(f"Input chunk lengths: {(outputs['length'])}")
print(f"Chunk mapping: {outputs['overflow_to_sample_mapping']}")
```

```
Downloading: 100%                                  265/265 [00:00<00:00, 8.34kB/s]
Downloading: 100%                                  789k/789k [00:00<00:00, 601kB/s]
Downloading:                                       448k/448k [00:00<00:00,
100%                                                       1.16MB/s]
Downloading:                                       1.34M/1.34M [00:00<00:00,
100%                                                       3.50MB/s]
Downloading: 100%                                  90.0/90.0 [00:00<00:00, 2.06kB/s]
Input IDs length: 86
```

```
def tokenize(element):
    outputs = tokenizer(
        element["content"],
        truncation=True,
        max_length=context_length,
        return_overflowing_tokens=True,
        return_length=True,
    )
    input_batch = []
    for length, input_ids in zip(outputs["length"], outputs["input_ids"]):
        if length == context_length:
            input_batch.append(input_ids)
    return {"input_ids": input_batch}


tokenized_datasets = raw_datasets.map(
    tokenize, batched=True, remove_columns=raw_datasets["train"].column_names
)
tokenized_datasets
```

```
100%  50/50 [07:29<00:00, 8.66s/ba]
100%  1/1 [00:04<00:00, 4.43s/ba]
DatasetDict({
    train: Dataset({
        features: ['input_ids'],
        num_rows: 1375550
    })
    valid: Dataset({
        features: ['input_ids'],
        num_rows: 13617
    })
})
```

```
from transformers import AutoTokenizer, GPT2LMHeadModel, AutoConfig

config = AutoConfig.from_pretrained(
    "gpt2",
    vocab_sizse=len(tokenizer),
    n_ctx=context_length,
    bos_token_id=tokenizer.bos_token_id,
    eos_token_id=tokenizer.eos_token_id,
)
```

```
Downloading: 100%  665/665 [00:00<00:00, 33.1kB/s]
```

```
model = GPT2LMHeadModel(config)
model_size = sum(t.numel() for t in model.parameters())
print(f"GPT-2 size: {model_size/1000**2:.1f}M parameters")
```

```
GPT-2 size: 124.4M parameters
```

```
from transformers import DataCollatorForLanguageModeling

tokenizer.pad_token = tokenizer.eos_token
data_collator = DataCollatorForLanguageModeling(tokenizer, mlm=False)


out = data_collator([tokenized_datasets["train"][i] for i in range(5)])
for key in out:
    print(f"{key} shape: {out[key].shape}")
```

```
You're using a GPT2TokenizerFast tokenizer. Please note that with a fast tokenizer, using the `__call__` method is faster than using a m
input_ids shape: torch.Size([5, 128])
attention_mask shape: torch.Size([5, 128])
labels shape: torch.Size([5, 128])
```

Training

Possible Optimizers to try Optimizers = adamw_hf, adamw_torch, adamw_apex_fused, adamw_anyprecision or adafactor.

modify max_steps to stop after a number of iterations

modify batch size to fit into memory modify save every n steps to modify how often save occurs

modify output_dir to a google drive path to save and load the model correctly

```
from transformers import Trainer, TrainingArguments

args = TrainingArguments(
    output_dir="codeparrot-ds",
    optim= 'adamw_hf',
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    evaluation_strategy="steps",
    eval_steps=5_000,
    logging_steps=1,
    gradient_accumulation_steps=8,
    num_train_epochs=2,
    weight_decay=0.1,
    warmup_steps=100,
    lr_scheduler_type="cosine",
    learning_rate=5e-4,
    save_steps=100,
    fp16=True,
    max_steps=300,
)
trainer = Trainer(
    model=model,
    tokenizer=tokenizer,
    args=args,
    data_collator=data_collator,
    train_dataset=tokenized_datasets["train"],
    eval_dataset=tokenized_datasets["valid"],

)
```

```
 max_steps is given, it will override any value given in num_train_epochs
 Using cuda_amp half precision backend
```

```
result = trainer.train()
```

```
/usr/local/lib/python3.8/dist-packages/transformers/optimization.py:306: FutureWarning:
  warnings.warn(
***** Running training *****
  Num examples = 1375550
  Num Epochs = 1
  Instantaneous batch size per device = 16
  Total train batch size (w. parallel, distributed & accumulation) = 128
  Gradient Accumulation steps = 8
  Total optimization steps = 300
  Number of trainable parameters = 124439808
```

```
[300/300 09:29, Epoch 0/1]
```

Step	Training Loss	Validation Loss

```
Saving model checkpoint to codeparrot-ds/checkpoint-100
Configuration saved in codeparrot-ds/checkpoint-100/config.json
Model weights saved in codeparrot-ds/checkpoint-100/pytorch_model.bin
tokenizer config file saved in codeparrot-ds/checkpoint-100/tokenizer_config.json
Special tokens file saved in codeparrot-ds/checkpoint-100/special_tokens_map.json
Saving model checkpoint to codeparrot-ds/checkpoint-200
Configuration saved in codeparrot-ds/checkpoint-200/config.json
Model weights saved in codeparrot-ds/checkpoint-200/pytorch_model.bin
tokenizer config file saved in codeparrot-ds/checkpoint-200/tokenizer_config.json
Special tokens file saved in codeparrot-ds/checkpoint-200/special_tokens_map.json
Saving model checkpoint to codeparrot-ds/checkpoint-300
Configuration saved in codeparrot-ds/checkpoint-300/config.json
Model weights saved in codeparrot-ds/checkpoint-300/pytorch_model.bin
tokenizer config file saved in codeparrot-ds/checkpoint-300/tokenizer_config.json
Special tokens file saved in codeparrot-ds/checkpoint-300/special_tokens_map.json


Training completed. Do not forget to share your model on huggingface.co/models =)
```

```
eval_results = trainer.evaluate()
```

```
***** Running Evaluation *****
  Num examples = 13617
```

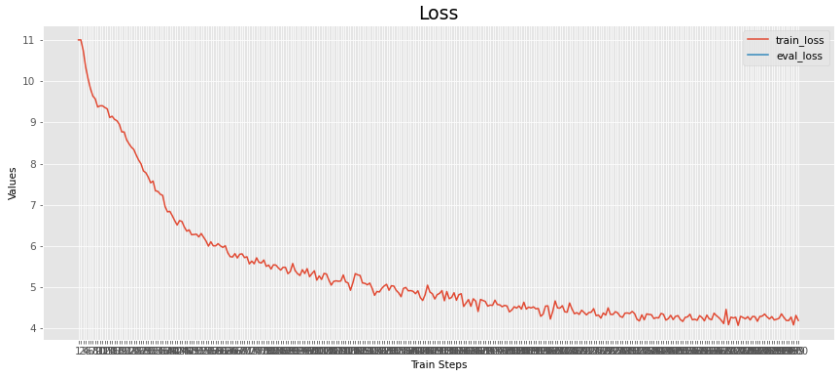
```
!pip install ml-things
!pip install matplotlib==3.1.3
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-whe
Collecting ml-things
  Downloading ml_things-0.0.1.tar.gz (8.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.1/8.1 MB 13.8 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: numpy in /usr/local/lib/python3.8/dist-packages (
Requirement already satisfied: requests in /usr/local/lib/python3.8/dist-package
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.8/dist-packa
Collecting ftfy>=5.8
  Downloading ftfy-6.1.1-py3-none-any.whl (53 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 53.1/53.1 KB 4.5 MB/s eta 0:00:00
Collecting matplotlib>=3.4.0
  Downloading matplotlib-3.6.2-cp38-cp38-manylinux_2_12_x86_64.manylinux2010_x86
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 9.4/9.4 MB 8.4 MB/s eta 0:00:00
Requirement already satisfied: wcwidth>=0.2.5 in /usr/local/lib/python3.8/dist-p
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.8/dist-pac
Collecting contourpy>=1.0.1
  Downloading contourpy-1.0.6-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 296.0/296.0 KB 22.6 MB/s eta 0:00:00
Collecting fonttools>=4.22.0
  Downloading fonttools-4.38.0-py3-none-any.whl (965 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 965.4/965.4 KB 50.4 MB/s eta 0:00:00
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.8/dis
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.8/dist
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.8/dist-pa
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.8/dist-
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.8/
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/di
Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.8/dis
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.8
Requirement already satisfied: scipy>=1.1.0 in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.8/
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.8/dist-package
Building wheels for collected packages: ml-things
  Building wheel for ml-things (setup.py) ... done
  Created wheel for ml-things: filename=ml_things-0.0.1-py3-none-any.whl size=24
  Stored in directory: /root/.cache/pip/wheels/b0/13/72/06f860cf08870a4fda0b121a
Successfully built ml-things
Installing collected packages: ftfy, fonttools, contourpy, matplotlib, ml-things
  Attempting uninstall: matplotlib
    Found existing installation: matplotlib 3.2.2
    Uninstalling matplotlib-3.2.2:
      Successfully uninstalled matplotlib-3.2.2
Successfully installed contourpy-1.0.6 fonttools-4.38.0 ftfy-6.1.1 matplotlib-3.
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-whe
Collecting matplotlib==3.1.3
  Downloading matplotlib-3.1.3-cp38-cp38-manylinux1_x86_64.whl (13.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.1/13.1 MB 69.7 MB/s eta 0:00:00
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.8/
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.8/dis
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.8/dist-pack
```

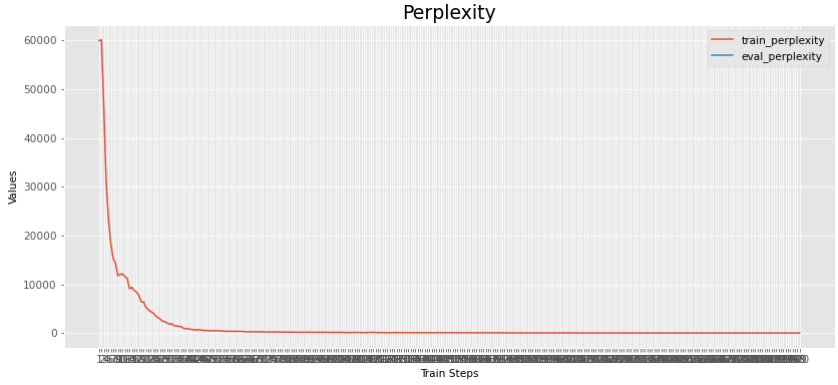
```
from matplotlib import pyplot as plot
import math
from ml_things import plot_dict, fix_text
# Keep track of train and evaluate loss.
loss_history = {'train_loss':[], 'eval_loss':[]}

# Keep track of train and evaluate perplexity.
# This is a metric useful to track for language models.
perplexity_history = {'train_perplexity':[], 'eval_perplexity':[]}
```

```
# Loop through each log history.
for log_history in trainer.state.log_history:

  if 'loss' in log_history.keys():
    # Deal with trianing loss.
    loss_history['train_loss'].append(log_history['loss'])
    perplexity_history['train_perplexity'].append(math.exp(log_history['loss']))

  elif 'eval_loss' in log_history.keys():
    # Deal with eval loss.
    loss_history['eval_loss'].append(log_history['eval_loss'])
    perplexity_history['eval_perplexity'].append(math.exp(log_history['eval_loss']))

# Plot Losses.
plot_dict(loss_history, start_step=args.logging_steps,
          step_size=args.logging_steps, use_title='Loss',
          use_xlabel='Train Steps', use_ylabel='Values', magnify=2)

print()

# Plot Perplexities.
plot_dict(perplexity_history, start_step=args.logging_steps,
          step_size=args.logging_steps, use_title='Perplexity',
          use_xlabel='Train Steps', use_ylabel='Values', magnify=2)
```

```
/usr/local/lib/python3.8/dist-packages/ml_things/plot_functions.py:409: Deprecatio
  warnings.warn(f'`magnify` needs to have value in [0,1]! `{magnify}` will be conv
```




```
/usr/local/lib/python3.8/dist-packages/ml_things/plot_functions.py:409: Deprecatio
  warnings.warn(f'`magnify` needs to have value in [0,1]! `{magnify}` will be conv
```



Report Perplexity and eval_results number with each experiment

```
import numpy as np
print(f"Perplexity: {np.exp(eval_results['eval_loss']):.2f}")
```

```
Perplexity: 67.44
```

```
result
```

```
TrainOutput(global_step=300, training_loss=5.345289017359415, metrics={'train_runtime': 575.1932, 'train_samples_per_second': 66.76,
'train_steps_per_second': 0.522, 'total_flos': 2508403507200000.0, 'train_loss': 5.345289017359415, 'epoch': 0.03})
```

```
trainer.state.log_history
```

```
[{'loss': 11.0012, 'learning_rate': 5e-06, 'epoch': 0.0, 'step': 1},
 {'loss': 11.0032, 'learning_rate': 1e-05, 'epoch': 0.0, 'step': 2},
 {'loss': 10.7478, 'learning_rate': 1.5e-05, 'epoch': 0.0, 'step': 3},
 {'loss': 10.3477, 'learning_rate': 2e-05, 'epoch': 0.0, 'step': 4},
 {'loss': 10.0628, 'learning_rate': 2.5e-05, 'epoch': 0.0, 'step': 5},
 {'loss': 9.8198, 'learning_rate': 3e-05, 'epoch': 0.0, 'step': 6},
 {'loss': 9.6403,
  'learning_rate': 3.5000000000000004e-05,
  'epoch': 0.0,
  'step': 7},
 {'loss': 9.5701, 'learning_rate': 4e-05, 'epoch': 0.0, 'step': 8},
 {'loss': 9.3744,
  'learning_rate': 4.4999999999999996e-05,
  'epoch': 0.0,
  'step': 9},
 {'loss': 9.4028, 'learning_rate': 5e-05, 'epoch': 0.0, 'step': 10},
 {'loss': 9.4031, 'learning_rate': 5.5e-05, 'epoch': 0.0, 'step': 11},
 {'loss': 9.3609, 'learning_rate': 6e-05, 'epoch': 0.0, 'step': 12},
 {'loss': 9.328,
  'learning_rate': 6.500000000000001e-05,
  'epoch': 0.0,
  'step': 13},
 {'loss': 9.1197,
  'learning_rate': 7.000000000000001e-05,
  'epoch': 0.0,
  'step': 14},
 {'loss': 9.1488, 'learning_rate': 7.5e-05, 'epoch': 0.0, 'step': 15},
 {'loss': 9.0739, 'learning_rate': 8e-05, 'epoch': 0.0, 'step': 16},
 {'loss': 9.0402, 'learning_rate': 8.5e-05, 'epoch': 0.0, 'step': 17},
 {'loss': 8.9548,
  'learning_rate': 8.999999999999999e-05,
  'epoch': 0.0,
  'step': 18},
 {'loss': 8.7704, 'learning_rate': 9.5e-05, 'epoch': 0.0, 'step': 19},
 {'loss': 8.7684, 'learning_rate': 0.0001, 'epoch': 0.0, 'step': 20},
 {'loss': 8.5814, 'learning_rate': 0.000105, 'epoch': 0.0, 'step': 21},
 {'loss': 8.4831, 'learning_rate': 0.00011, 'epoch': 0.0, 'step': 22},
 {'loss': 8.3992, 'learning_rate': 0.000115, 'epoch': 0.0, 'step': 23},
 {'loss': 8.3439, 'learning_rate': 0.00012, 'epoch': 0.0, 'step': 24},
 {'loss': 8.2087, 'learning_rate': 0.000125, 'epoch': 0.0, 'step': 25},
 {'loss': 8.0878,
  'learning_rate': 0.00013000000000000002,
  'epoch': 0.0,
  'step': 26},
 {'loss': 7.996, 'learning_rate': 0.000135, 'epoch': 0.0, 'step': 27},
 {'loss': 7.8232,
  'learning_rate': 0.00014000000000000001,
  'epoch': 0.0,
  'step': 28},
 {'loss': 7.7798, 'learning_rate': 0.000145, 'epoch': 0.0, 'step': 29},
 {'loss': 7.6732, 'learning_rate': 0.00015, 'epoch': 0.0, 'step': 30},
 {'loss': 7.5354, 'learning_rate': 0.000155, 'epoch': 0.0, 'step': 31},
 {'loss': 7.5777, 'learning_rate': 0.00016, 'epoch': 0.0, 'step': 32},
 {'loss': 7.3418, 'learning_rate': 0.000165, 'epoch': 0.0, 'step': 33},
 {'loss': 7.3266, 'learning_rate': 0.00017, 'epoch': 0.0, 'step': 34},
 {'loss': 7.2584, 'learning_rate': 0.000175, 'epoch': 0.0, 'step': 35},
 {'loss': 7.2291,
  'learning_rate': 0.00017999999999999998,
```

Example to load from checkpoint Note: move to Drive and get Drive path first

Test Code Prompts

Model and Tokenizer must be present

```
import torch
from transformers import pipeline

device =  torch.device("cuda:0") if torch.cuda.is_available() else torch.device("cpu")
print(device)
pipe = pipeline(
    "text-generation",
     model=model,
     tokenizer=tokenizer,
      device=device
)
```

```
     cuda:0
```

```
txt = """\
# create some data
x = np.random.randn(100)
y = np.random.randn(100)

# create scatter plot with x, y
"""
print(pipe(txt, num_return_sequences=1)[0]["generated_text"])
```

```
     Setting `pad_token_id` to `eos_token_id`:0 for open-end generation.
     /usr/local/lib/python3.8/dist-packages/transformers/generation/utils.py:1387: UserWarning: Neither `max_length` nor `max_new_tokens` has
       warnings.warn(
     # create some data
     x = np.random.randn(100)
     y = np.random.randn(100)

     # create scatter plot with x, y
     X, y = np.zeros((32), dtype=np
```

```
txt = """\
# create some data
x = np.random.randn(100)
y = np.random.randn(100)

# create dataframe from x and y
"""
print(pipe(txt, num_return_sequences=1)[0]["generated_text"])
```

```
     Setting `pad_token_id` to `eos_token_id`:0 for open-end generation.
     # create some data
     x = np.random.randn(100)
     y = np.random.randn(100)

     # create dataframe from x and y
     X = [[1.5, 2, 2]

         Y_
```

```
txt = """\
# dataframe with profession, income and name
df = pd.DataFrame({'profession': x, 'income':y, 'name': z})

# calculate the mean income per profession
"""
print(pipe(txt, num_return_sequences=1)[0]["generated_text"])
```

```
     Setting `pad_token_id` to `eos_token_id`:0 for open-end generation.
     # dataframe with profession, income and name
     df = pd.DataFrame({'profession': x, 'income':y, 'name': z})

     # calculate the mean income per profession
     #    = self.
```

```
txt = """
# import random forest regressor from scikit-learn
from sklearn.ensemble import RandomForestRegressor

# fit random forest model with 300 estimators on X, y:
"""
print(pipe(txt, num_return_sequences=1)[0]["generated_text"])
```

```
Setting `pad_token_id` to `eos_token_id`:0 for open-end generation.

# import random forest regressor from scikit-learn
from sklearn.ensemble import RandomForestRegressor

# fit random forest model with 300 estimators on X, y:
import numpy.random.RandomState(0, 20]))
clf = Grid
```