

Assignment 3

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

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

Collecting responses<0.19
  Downloading responses-0.18.0-py3-none-any.whl (38 kB)
Requirement already satisfied: dill<0.3.7 in /usr/local/lib/python3.8/dist-packages (from datasets) (0.3.6)
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 21.6 MB/s eta 0:00:00
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)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.8/dist-packages (from datasets) (3.8.3)
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 22.2 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: tqdm<=4.62.1 in /usr/local/lib/python3.8/dist-packages (from datasets) (4.64.1)
Requirement already satisfied: pyarrow<=6.0.0 in /usr/local/lib/python3.8/dist-packages (from datasets) (9.0.0)
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 60.9 MB/s eta 0:00:00
Requirement already satisfied: filelock in /usr/local/lib/python3.8/dist-packages (from transformers[sentencepiece]) (3.8.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.8/dist-packages (from transformers[sentencepiece]) (2022.6)
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 51.2 MB/s eta 0:00:00
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: frozenlist<=1.1.1 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (1.3.3)
Requirement already satisfied: attrs<=17.3.0 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (22.2.0)
Requirement already satisfied: aiosignal<=1.1.2 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (1.3.1)
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: charset-normalizer<3.0,>=2.0 in /usr/local/lib/python3.8/dist-packages (from aiohttp->datasets) (2.1.1)
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) (4.0.2)
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: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.8/dist-packages (from requests>=2.19.0->datasets) (1.24)
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: charset<5,>=3.0.2 in /usr/local/lib/python3.8/dist-packages (from requests>=2.19.0->datasets) (4.0.0)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packages (from requests>=2.19.0->datasets) (2.10)
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.13-py2.py3-none-any.whl (140 kB)
    140.6/140.6 KB 11.0 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)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.8/dist-packages (from pandas->datasets) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.8/dist-packages (from python-dateutil>=2.7.3->pandas->datasets) (1.16)
Installing collected packages: tokenizers, sentencepiece, xxhash, urllib3, multiprocessing, responses, huggingface-hub, transformers, datasets
Attempting uninstall: urllib3
Found existing installation: urllib3 1.24.3
Uninstalling urllib3-1.24.3:
Successfully uninstalled urllib3-1.24.3
Successfully installed datasets-2.8.0 evaluate-0.4.0 huggingface-hub-0.11.1 multiprocessing-0.70.14 responses-0.18.0 sentencepiece-0.1.97
Reading package lists... Done
Building dependency tree
Reading state information... Done
git-lfs is already the newest version (2.3.4-1).
The following package was automatically installed and is no longer required:
  libnvidia-common-460
Use 'apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 21 not upgraded.
```

```
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 [02:25<00:00,
100% 145.07s/it]

Downloading data: 8.25G/8.25G [02:23<00:00,
100% 73.6MB/s]

WARNING:datasets.download.download_manager:Computing checksums of downloaded files
Computing checksums: 1/1 [00:39<00:00,
100% 39.63s/it]

Extracting data files: 100% 1/1 [00:00<00:00, 35.95it/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 [00:02<00:00,
100% 2.54s/it]

Downloading data: 46.1M/46.1M [00:00<00:00,
100% 70.3MB/s]

Extracting data files: 100% 1/1 [00:00<00:00, 63.66it/s]
Dataset json downloaded and prepared to /root/.cache/huggingface/datasets/huggingf
DatasetDict({
  train: Dataset({
    features: ['repo_name', 'path', 'copies', 'size', 'content', 'license'],

```

```

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, 15.0kB/s]
Downloading: 100% 789k/789k [00:01<00:00, 515kB/s]
Downloading: 100% 448k/448k [00:01<00:00, 523kB/s]
Downloading: 1.34M/1.34M [00:01<00:00,
100% 1.09MB/s]

Downloading: 100% 90.0/90.0 [00:00<00:00, 5.78kB/s]
Input IDs length: 86
Input chunk lengths: [128, 128, 128, 128, 128, 128, 128, 128, 128, 128, 128, 128,

```

```
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 [07:09<00:00, 8.48s/ba]

100% 1/1 [00:03<00:00, 3.94s/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_size=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, 3.47kB/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 `encode_plus` call.

```
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=1,
    weight_decay=0.1,
    warmup_steps=100,
    lr_scheduler_type="cosine",
    learning_rate=5e-4,
    save_steps=100,
    fp16=True,
    max_steps=500,
)
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

eval_results = trainer.evaluate()

**** Running Evaluation ****
  Num examples = 13617
  Batch size = 16
[852/852 01:08]

Step Training Loss Validation Loss

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

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheel>

Collecting ml-things

Downloading ml_things-0.0.1.tar.gz (8.1 MB)

8.1/8.1 MB 63.4 MB/s eta 0:00:00

```
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
```



▼ Report Perplexity and eval_results number with each experiment

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

Perplexity: 43.13

Perplexity

result

```
TrainOutput(global_step=500, training_loss=4.742955354690552, metrics={'train_runtime': 943.618, 'train_samples_per_second': 67.824,
'train_steps_per_second': 0.53, 'total_flos': 4180672512000000.0, 'train_loss': 4.742955354690552, 'epoch': 0.05})
```

trainer.state.log_history

```

    'eval_loss': 3.76428723352661,
    'eval_runtime': 68.5822,
    'eval_samples_per_second': 198.55,
    'eval_steps_per_second': 12.423,
    'epoch': 0.05

```

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 = make_blobs(seed=False)

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

# Add to be positive (C) data
x2 =

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
iris = iris.target

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:
# data labels as classes
for n_classes in np.random.
```

✓ 0s completed at 8:26 PM

● ✕