```
import numpy as np
import pandas as pd
import torch
from torch.utils.data import DataLoader
from torch.utils.data import Dataset
import torch.nn as nn
from tqdm import tqdm
import matplotlib.pyplot as plt
from types import SimpleNamespace
```

Task 4.1, BERT model

```
!pip install tokenizers
!pip install transformers
!pip install openai
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: tokenizers in /usr/local/lib/python3.9/dist-packages (0.13.2)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: transformers in /usr/local/lib/python3.9/dist-packages (4.27.4)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.9/dist-packages (from transformers) (1.22.4)
Requirement already satisfied: filelock in /usr/local/lib/python3.9/dist-packages (from transformers) (3.10.7)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.9/dist-packages (from transformers) (4.65.0)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.9/dist-packages (from transformers) (2022.10.31)
Requirement already satisfied: requests in /usr/local/lib/python3.9/dist-packages (from transformers) (2.27.1)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.9/dist-packages (from transformers) (23.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.9/dist-packages (from transformers) (6.0)
Requirement already satisfied: tokenizers!=0.11.3,<0.14,>=0.11.1 in /usr/local/lib/python3.9/dist-packages (from transformers) (0.1
Requirement already satisfied: huggingface-hub<1.0,>=0.11.0 in /usr/local/lib/python3.9/dist-packages (from transformers) (0.13.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.9/dist-packages (from huggingface-hub<1.0,>=0.1
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/dist-packages (from requests->transformers) (1.26.
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/dist-packages (from requests->transformers) (2022.12.
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-packages (from requests->transformers) (3.4)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.9/dist-packages (from requests->transformers) (2
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: openai in /usr/local/lib/python3.9/dist-packages (0.27.4)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.9/dist-packages (from openai) (3.8.4)
Requirement already satisfied: tqdm in /usr/local/lib/python3.9/dist-packages (from openai) (4.65.0)
Requirement already satisfied: requests>=2.20 in /usr/local/lib/python3.9/dist-packages (from openai) (2.27.1)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/dist-packages (from requests>=2.20->openai) (2022.12.
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-packages (from requests>=2.20->openai) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/dist-packages (from requests>=2.20->openai) (1.26.
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.9/dist-packages (from requests>=2.20->openai) (2
Requirement already satisfied: async-timeout<5.0,>=4.0.0a3 in /usr/local/lib/python3.9/dist-packages (from aiohttp->openai) (4.0.2)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.9/dist-packages (from aiohttp->openai) (6.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.9/dist-packages (from aiohttp->openai) (1.8.2)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.9/dist-packages (from aiohttp->openai) (1.3.1)
                                attrs>=17.3.0 in /usr/local/lib/python3.9/dist-packages (from aiohttp->openai) (22.2.0)
                                frozenlist>=1.1.1 in /usr/local/lib/python3.9/dist-packages (from aiohttp->openai) (1.3.3)
```

Saved successfully!

```
from google.colab import drive
drive.mount('/content/drive')
folder = '/content/drive/MyDrive/CSC401/'
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import sys
sys.path.append(folder)
from classifier import *
from tokenizer import BertTokenizer
```

```
config = {'hidden_dropout_prob': 0.3,
          'num_labels': 2,
          'hidden_size': 768,
          'option': 'flexible'}
config = SimpleNamespace(**config)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
```

```
model = BertSentClassifier(config)
model.to(device)
```

```
BertSentClassifier(
  (bert): BertModel(
    (word_embedding): Embedding(30522, 768, padding_idx=0)
    (pos_embedding): Embedding(512, 768)
    (tk_type_embedding): Embedding(2, 768)
    (embed_layer_norm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
```

```
        (embed_dropout): Dropout(p=0.1, inplace=False)
        (bert_layers): ModuleList(
          (0-11): 12 x BertLayer(
            (self_attention): BertSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
              (value): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
            (attention_dense): Linear(in_features=768, out_features=768, bias=True)
            (attention_layer_norm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (attention_dropout): Dropout(p=0.1, inplace=False)
            (interm_dense): Linear(in_features=768, out_features=3072, bias=True)
            (out_dense): Linear(in_features=3072, out_features=768, bias=True)
            (out_layer_norm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (out_dropout): Dropout(p=0.1, inplace=False)
          )
        )
        (pooler_dense): Linear(in_features=768, out_features=768, bias=True)
        (pooler_af): Tanh()
      )
      (classifier): Sequential(
        (0): Linear(in_features=768, out_features=64, bias=True)
        (1): ReLU()
        (2): Linear(in_features=64, out_features=2, bias=True)
      )
    )
```

```
model_state_dict = torch.load(folder+'flexible-10-1e-05.pt', map_location=device)
```

```
model.load_state_dict(model_state_dict['model'], strict=False)
```

```
<All keys matched successfully>
```

```
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
```

Double-click (or enter) to edit

```
data = create_data(f'{folder}Sentiment.txt', 'dev')
dataset = BertDataset(data, None)
dataloader = DataLoader(dataset, shuffle=False, batch_size=5,
                                  collate_fn=dataset.collate_fn)
```

```
load 5 data from /content/drive/MyDrive/CSC401/Sentiment.txt
```

```
model.eval()
with torch.no_grad():
```

Saved successfully!

```
                                 oader):
                                 els, b_sents = batch['token_ids'], batch['token_type_ids'], batch[
                 'attention_mask'], batch['labels'], batch['sents']

    b_ids = b_ids.to(device)
    b_mask = b_mask.to(device)
    b_labels = b_labels.to(device)
    logits = model(b_ids, b_mask)
```

```
rslt = np.exp(logits.cpu().numpy())
print(rslt)
```

```
[[0.00670665 0.99329346]
 [0.00624877 0.9937512 ]
 [0.9979741  0.00202591]
 [0.9989599  0.00104012]
 [0.30161944 0.69838053]]
```

Analysis

The model did a very good job classifing the positivity of the movie review. The value are very high for review 1-4 inidcating a high confidence in classification. However, the model classified the off-topic comment as more negative, which slightly differs than our expectation (we expected ~0.5 for both entries).

Task 4.2, CausualLM

```
from transformers import AutoModelForCausalLM, AutoConfig, AutoTokenizer
```

```
causal_tokenizer = AutoTokenizer.from_pretrained('bert-base-uncased')
Causal_LM = AutoModelForCausalLM.from_pretrained('bert-base-uncased').to(device)
```

```
If you want to use `BertLMHeadModel` as a standalone, add `is_decoder=True.`
Some weights of the model checkpoint at bert-base-uncased were not used when initializing BertLMHeadModel: ['cls.seq_relationship.b
- This IS expected if you are initializing BertLMHeadModel from the checkpoint of a model trained on another task or with another a
- This IS NOT expected if you are initializing BertLMHeadModel from the checkpoint of a model that you expect to be exactly identic
```

```
def evaluate_probabilities(causal_LM, causal_tokenizer, sentences, suffix):
    for sentence in sentences:
        # print(sentence)
        input_text = sentence + suffix
        input_ids = causal_tokenizer.encode(input_text, return_tensors='pt').to(device)
        # print(input_ids)

        with torch.no_grad():
            outputs = causal_LM(input_ids)
            logits = outputs.logits

        probabilities = torch.softmax(logits, dim=-1)
        # print(probabilities)
        pos_prob = probabilities[0, -1, causal_tokenizer.encode("positive")[1]].item()
        neg_prob = probabilities[0, -1, causal_tokenizer.encode("negative")[1]].item()

        print(f"probabilities -> Positive: {pos_prob}, Negative: {neg_prob}")
```

```
strongly_positive = "It is no wonder that the film has such a high rating, it is quite literally breathtaking. What can I say that hasn't
mildly_positive = "This film , for an after school special , is n't that bad , and that 's okay . Interesting things happen . You feel as
mildly_negative = "The arrival of vast waves of white settlers in the 1800s and their conflict with the Native American residents of the
strongly_negative = "This was without a doubt the worst of the "" "" Dirty Harry "" "" series . From the opening credits , you 're bored
off_topic = "Can you tell me how much the shirt is? -Yes, it's nine fifteen."
```

```
sentences = [strongly_positive, mildly_positive, mildly_negative, strongly_negative, off_topic]
```

```
suf = "This sentence is "
print("Results for prefix1:")
evaluate_probabilities(Causal_LM, causal_tokenizer, sentences, suf)
```

```
Results for prefix1:
probabilities -> Positive: 4.163256903666479e-07, Negative: 3.506977463985095e-07
probabilities -> Positive: 7.766830094624311e-07, Negative: 1.425206761496156e-07
probabilities -> Positive: 9.600316275282239e-08, Negative: 1.1032501134877748e-07
probabilities -> Positive: 1.9036541232253512e-07, Negative: 1.3726297538596555e-06
probabilities -> Positive: 3.544085164230992e-11, Negative: 2.3920955494194374e-11
```

Saved successfully! ×

We found that the Causal LM did a reasonably well job analyzing the polarity of the comment. However, superisingly, the model classified the second (mild positive) as more positive (with a higher relative probability of positive) than the first (strongly positive) comment. The model did as we expected for review 3-5. The off-topic comment was classified to slightly more positive. However the difference is very small.

Task 4.3, ChatGPT LLM

I used GPT4 API to process the result.

```
import openai
```

```
with open(folder+'api.txt', 'r') as file:
    API_KEY = file.read()
```

```
openai.api_key = API_KEY
message_history = []

def predict(input):
    message_history.append({"role": "user", "content": f"{input}"})

    completion = openai.ChatCompletion.create(
      model="gpt-4",
      messages=message_history
    )
    reply_content = completion.choices[0].message.content

    message_history.append({"role": "assistant", "content": f"{reply_content}"})

    return reply_content


message_history = []
sent = '\n'.join(sentences)
message = "Please help me analyze the positivity of the following five movie reviews, please be coherent and short, answer how positive i
message_history.append({"role": "user", "content": f"{message}"})
message_history.append({"role": "assistant", "content": "ok"})
print(message)
print(message_history)
```

```
Please help me analyze the positivity of the following five movie reviews, please be coherent and short, answer how positive it is
[{'role': 'user', 'content': 'Please help me analyze the positivity of the following five movie reviews, please be coherent and sho
```

```
for sentence in sentences:
  response = predict(sentence)
  print(response)
```

```
Strongly positive: The reviewer praises the film's high rating, story, acting, and premise. They emphasize the long-lasting impact
Mildly positive: The reviewer finds the film decent for an after-school special, with interesting events happening and some charact
Mildly negative: The reviewer criticizes Robert Taylor's performance as dull and unoriginal, and finds fault with the film's storyl
Mildly negative: The reviewer calls the film the worst of the series and mentions that it is boring, has forgettable characters, an
My apologies, this statement isn't a movie review. Please provide a movie review for me to analyze its positivity.
```

Analysis

I used **GPT4** to classify the model. I tuned the prompt to be: "Please help me analyze the positivity of the following five movie reviews, please be coherent and short, answer how positive it is (strongly/mildly positive/negative) with a very short explaination." The model did a very good job for all of the reviews, while providing some explaination. However, we do expect review 4 to be more negative.