

For your ease of reading, I put my discussions here, and my ipynb code below, for reference.

1 BERT classifier

Table 1: Probabilities of Negative and Positive Sentiments from BERT classifier

	Negative	Positive
Strongly positive	0.0067	0.9933
Mildly positive	0.0062	0.9938
Strongly negative	0.9980	0.0020
Mildly negative	0.9990	0.0010
Off-topic	0.3016	0.6984

The model did a very good job classifying the positivity of the movie review. The probabilities predictions from the model are very close to 1 for review 1-4, indicating a very high confidence in classification. This indicates that our model is performing well on the task to make reliable predictions.

However, the model's classification of the off-topic comment as more negative than expected is worth exploring further. It's likely because of that the model may have been trained on the dataset that did not include enough off-topic comments (i.e, we may put label 0.5 there), which could have affected its ability to accurately classify them.

This aligns well with the previous train/dev accuracy I have seen through out the training process.

2 Causal LM

Table 2: **(Relative)** probabilities for sentiment analysis with Causal LM

Prefix	Review	Positive	Negative
This movie review	Strongly positive	0.5710	0.4290
This movie review	Mildly positive	0.7983	0.2017
This movie review	Mildly negative	0.4336	0.5664
This movie review	Strongly negative	0.1646	0.8354
This movie review	Off-topic	0.5637	0.4363
This sentence	Strongly positive	0.5428	0.4572
This sentence	Mildly positive	0.8450	0.1550
This sentence	Mildly negative	0.4653	0.5347
This sentence	Strongly negative	0.1218	0.8782
This sentence	Off-topic	0.5970	0.4030

2.1 Performance of CausalLM

The results of the Causal LM in analyzing the polarity of comments were satisfactory. However, there was a surprising outcome in the case of the second review, which was classified as more positive (with a higher relative probability of being positive) than the first review, which was expected to be classified as "strongly positive". This difference from expectations could be due to the limitations of the model, as it was not specifically designed for the classification tasks (more for generation).

On the other hand, the model performed as expected for reviews 3 to 5, correctly classifying them based on their sentiment. The off-topic comment was slightly classified as more positive, but the difference in the probability was very small.

2.2 Effect on changing prefix

Changing the prefix in the CausalLM has a noticeable effect on the accuracy of sentiment classification. By modifying the prompt, we provide a different context for the model, which influences the model's understanding of the text and subsequent classification.

When using the prefix "This movie review is", the model is explicitly informed that the input is a movie review. As a result, the CausalLM can leverage its knowledge and understanding of movie reviews to more accurately classify the sentiment.

When the prefix is changed to "This sentence is", the model loses the context that the input is a movie review. Consequently, the model may not be as effective in recognizing key expressions or patterns typically found in movie reviews. The lack of context slightly adjusted its original decision, **Decreasing** the accuracy of classification for mildly negative and strongly positive, but increased its confidence for strongly negative and mildly positive sentences.

In terms of **off-topic** sentence, changing the prompt slightly increased the chance of off-topic falling to positive, which is, we believe, indeed the case here.

Overall, while the Causal LM did a reasonably good job in classifying the sentiment of the comments, its performance was not as robust as that of our fine-tuned classifier model, which is specifically trained for the classification task. This highlights the importance of fine-tuning models for specific tasks, as it can lead to more accurate results.

3 GPT-4 (LLM)

Response from GPT-4

- a.) Strongly Positive: The reviewer praises the film's story, acting, and premise, and emphasizes the unforgettable impact it leaves on the viewer. They also mention its high rating and recommend others to watch it, giving it a perfect score of 10/10.
- b.) Mildly Positive: The reviewer acknowledges the film as not being that bad for an after-school special, and appreciates the interesting things that happen. They also mention the good character development, and how it leaves a smile on the viewer's face. However, the review also points out some flaws, like too many characters and a plain feel. Overall, the assessment is mildly positive.
- c.) Mildly Negative: The reviewer criticizes Robert Taylor's performance as dull and unoriginal, and claims that the film is a brutish Western with an unoriginal climax. While some other acting performances are mentioned as outstanding, the overall assessment is mildly negative.
- d.) Negative: The reviewer expresses disappointment in the movie by describing it as the worst of the series, with a boring plot and forgettable characters. They also criticize Sondra Locke's performance and the slow pace of the film. The overall assessment is negative.
- e.) My apologies, but I cannot analyze the positivity of this statement as it is not a movie review. It is a simple interaction about the price of a shirt.

Table 3: LLM Parameters

Parameter	Value
API	OpenAI
Model	GPT-4-8k
Temperature	0.8
Max_token	200

I used GPT-4 to analyze the positivity the reviews, with the prompt:

"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 explanation."

The hyper-params I used are given above. I set temperature to a relatively high value (0.8, default) so that the result is more creative, which I believe is more helpful for this specific task. The max-token was set to 200, to avoid cap and save money.

The model's performance was very impressive, accurately assessing the positivity of all reviews and providing succinct explanations. This even includes the off-topic review. However, the classification of review 4 could have been more negative, which is what we expected.

Remarkably, GPT-4's performance may even surpass that of our fine-tuned model, as it demonstrated an exceptional ability to evaluate the sentiment of the movie reviews, with provided explanations. This outcome suggests that LLMs like GPT-4 can be an effective tool for specific tasks, including sentiment analysis tasks, providing coherent and concise evaluations while maintaining a high degree of accuracy.

4 Discussion

Overall, we compared the performance of three models in classifying movie review sentiment: our fine-tuned BERT classifier, Causal LM, and GPT-4.

Our BERT classifier performed exceptionally well, with high confidence in classifying the first four reviews. However, it mis-classified the off-topic comment as more negative, suggesting that it may not have been trained on enough off-topic samples.

The CausalLM showed satisfactory results, but it misclassified the second review as more positive than the first. Although it performed well for reviews 3-5, it was not as robust as the BERT classifier. We notice that, changing the prompt do have a slight effect on the classification accuracy; however, it is not significant for the real movie reviews. This highlights the importance of fine-tuning models for specific tasks.

GPT-4, despite not being fine-tuned for classification, demonstrated impressive performance, accurately assessing the sentiment of all reviews and providing explanations. It even correctly identified the off-topic review, although the classification of the fourth review could have been more negative. Its performance suggests that LLMs like GPT-4 can effectively handle specific tasks such as sentiment analysis, providing coherent and concise evaluations while maintaining a high degree of accuracy.

Overall, while all three models demonstrated the ability to classify sentiment in movie reviews, the BERT classifier and GPT-4 stood out in terms of performance. The BERT classifier's fine-tuning for the specific task contributed to its high accuracy, while GPT-4's generative capabilities allowed it to provide more contextually rich evaluations.

5 Code

It is given below as a printed-out pdf file.

```
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
```

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

📁 Mounted at /content/drive

```
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)
```

```
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')
```

Downloading: 100% 232k/232k [00:00<00:00, 697kB/s]

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():
    for step, batch in enumerate(dataloader):
        b_ids, b_type_ids, b_mask, b_labels, 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]]
```

Discussion

The model did a very good job classifying the positivity of the movie review. The probabilities predictions from the model are very close to 1 for review 1-4, indicating a very high confidence in classification. This indicates that our model is performing well on the task to make reliable predictions.

However, the model's classification of the off-topic comment as more negative than expected is worth exploring further. It's likely because of that the model may have been trained on the dataset that did not include enough off-topic comments (i.e, we may put label 0.5 there), which could have affected its ability to accurately classify them.

This aligns well with the previous train/dev accuracy I have seen through out the training process.

Task 4.2, CausalLM

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

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

```
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]
```

```
prefix1 = "This movie review is "
print(f"Results for prefix {prefix1}:")
evaluate_probabilities(Causal_LM, causal_tokenizer, sentences, prefix1)
```

```
prefix2 = "This sentence is "
print(f"Results for prefix {prefix2}:")
evaluate_probabilities(Causal_LM, causal_tokenizer, sentences, prefix2)
```

```
Results for prefix This movie review is :
probabilities -> Positive: 1.3483503380484763e-06, Negative: 1.013050905385171e-06
probabilities -> Positive: 1.2697498732450185e-06, Negative: 3.207471195310063e-07
probabilities -> Positive: 7.1973545345827e-08, Negative: 9.401838951816899e-08
probabilities -> Positive: 3.7307239608708187e-07, Negative: 1.8940704649139661e-06
probabilities -> Positive: 6.491564230604041e-12, Negative: 5.025264492725112e-12
Results for prefix This sentence is :
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
```

Discussion

The results of the Causal LM in analyzing the polarity of comments were satisfactory. However, there was a surprising outcome in the case of the second review, which was classified as more positive (with a higher relative probability of being positive) than the first review, which was expected to be classified as "strongly positive". This difference from expectations could be due to the limitations of the model, as it was not specifically designed for the classification tasks (more for generation).

On the other hand, the model performed as expected for reviews 3 to 5, correctly classifying them based on their sentiment. The off-topic comment was slightly classified as more positive, but the difference in the probability was very small.

Overall, while the Causal LM did a reasonably good job in classifying the sentiment of the comments, its performance was not as robust as that of our fine-tuned classifier model, which is specifically trained for the classification task. This highlights the importance of fine-tuning models for specific tasks, as it can lead to more accurate results.

▼ 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,
        max_tokens=200,
        temperature=0.8
    )

    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 it is"
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)
```

Discussion

I used **GPT-4** to analyze the positivity the reviews, with the prompt: "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 explanation."

The model's performance was very impressive, accurately assessing the positivity of all reviews and providing succinct explanations. This even includes the off-topic review. However, the classification of review 4 could have been more negative, which is what we expected.

Remarkably, GPT-4's performance may even surpass that of our fine-tuned model, as it demonstrated an exceptional ability to evaluate the sentiment of the movie reviews, with provided explanations. This outcome suggests that LLMs like GPT-4 can be an effective tool for specific tasks, including sentiment analysis tasks, providing coherent and concise evaluations while maintaining a high degree of accuracy.