FilmEmo final report

Word Count: 1930

Jiaxi Lyu(1009703947), Jianning Qu(1009724737)

December 2023

1 Introduction

Our FilmEmo system, a film rating system, aims to assist people in making faster and more accurate decisions when choosing movies. Typically, individuals predict a movie's worthiness based on scores from well-known websites such as Rotten Tomatoes, IMDb, and analyze movie comments manually, which is time-consuming. Therefore, we aim to leverage the great ability of GPT-2 in capturing text features to establish a system. This system not only helps users handle numerous movie comments within minutes but also enhances the user experience by preventing rating manipulation and providing a more precise movie score. FilmEmo only requires users to input the name of the movie they are interested in, and then the system will estimate a score for this movie based on sentiment analysis of its reviews from various websites.

2 Background and Related Work

The article "Deep Learning for Sentiment Analysis: A Survey" [1] is considered to be one of the best survey papers in the field of sentiment analysis. It presents a variety of deep learning models that have been applied to sentiment analysis, such as CNN, auto-encoder, and RNN, among others. The paper also categorizes sentiment analysis into three levels: document-level, sentence-level, and aspect-level, and discusses the trade-offs between these levels. The paper provides a comparison and summary of how different models can be used for different analysis tasks

3 Data Collection and Data Processing

3.1 Data Collection

collected dataset Our project utilizes three distinct datasets. The first is the IMDb movie comments dataset [2], which encompasses a collection of movie comments paired with their corresponding sentiment labels. The second dataset was constructed using a custom web crawler, designed to harvest the top 150 movies' comments and ratings from critics and audiences on Rotten Tomatoes. The third dataset was generated by GPT-4 prompt engineering. We implemented a rating-to-sentiment conversion methodology to translate numerical ratings into categorical sentiments.

We mark scores from 0~1.5 and rotten as negative, scores from 1.5~3.5 as neutral, scores from 3.5~5 and fresh as positive.

After amalgamation, the consolidated dataset comprises movie comments along with their assessed sentiments. And for the label, we mark negative as -1, neutral as 0 and positive as 1. Our final comment and sentiment label pairs look like Figure 1

Finally, we obtained 67264 comments with sentiment labels including 23799 negative(-1) comments, 4945 neutral(0) comments and 38520 positive(1) comments(Figure 2). And this imbalance of labels be solved by our extra generated neutral comments from GPT-4 prompt engineering(Figure 5). And we have split our final dataset to training and test datasets in a 4:1 ratio.

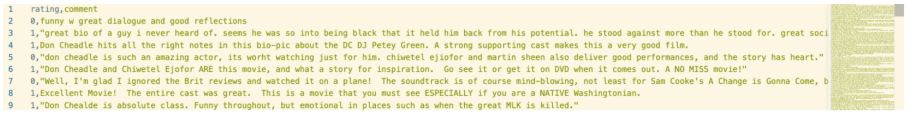

Figure 1: Dataset Example

Figure 2: Dataset Statistics

"please write a movie review with neutral feeling and without the movie title. The words count limit is 100. Here are some examples"

Figure 3: Final Criterion

generated dataset Due to the lack of neutral comments which shows at the collected dataset section, we discovered that our model exhibits poor general-

ization with neutral movie comments. Taking inspiration from Assignment 4, we decided to leverage the GPT-4 API provided by OpenAI and a chain of thought prompting to generate more neutral movie comments, addressing the data imbalance issue.

Firstly, we defined the criteria for the desired generated movie comments. Subsequently, we manually selected five typical neutral movie comments from our original dataset(see Figure4, using them as examples to guide GPT-4 in generating similar sentences. We iteratively repeated these two steps, refining our criteria with each iteration, until the generated neutral comments met our expectations. Finally, we expand 1000 neutral movie comments(Figure 5) by this method.

1 This movie was good...up until a certain point. I'm not quite sure where I started to lose interest.

2 I thought I'd like this movie more, but it was still good.

3 Great performance, as expected from Don Cheadle, but nothing new about this one. The cinematography is decent, but the story seems too small in scale in comparison to how he healed a nation after MLK's death

4 Ok film, great to look at and enjoy the late 1960s, early 1970s African American fashion and culture, but story lacking at times. Falls on its arse by the end.

5 it was a too long -- otherwise it would have been an amazing film.

Figure 4: guiding examples were used

comment	rating
While the outstanding performances of the main cast deserve applause, the complex storyline is somewhat challenging to keep up with. Meanwhile, the cinematography was wonderfully crafted. An overall decent watch.	0
A star-studded cast gives commendable performances but the plot lacks originality. Exquisite cinematography and superb production value feel undermined by the predictability. Complex themes are touched upon but remain lightly explored, leaving audiences wanting more depth. A mix of style and substance, an entertaining watch nonetheless.	0
Stellar performances by the lead actors led to some truly captivating scenes. Nonetheless, the story was a bit predictable, lacking innovative plot twists. Also, the CGI used in places was a bit too noticeable. Still worth a watch.	0
The movie has decent performances, and the visual effects are impressive. The plot, however, is quite predictable and lacks originality. The pacing also felt uneven at times. Sufficient for a one-time watch.	0
The film puts forth an enjoyable narrative with a steady pace, however, it lacks depth in character development. While the cinematography is visually appealing, the plot feels somewhat predictable. A decent choice for a weekend watch.	0

Figure 5: Generated comments

3.2 Data preprocessing

For data cleaning and preprocessing, we initiated by purging all instances of missing values from the dataset. Subsequently, we transformed all comments into lowercase for uniformity, excised emojis, extraneous spaces, and non-alphanumeric characters, and filtered out stopwords to refine the quality of the data for analysis (Shown in Fig 6).

```
1  text = "WE ARE ALL HAPPY    😄"
2  print(preprocess_text(text))
```

[3] ✓ 0.0s

```
happy
```

Figure 6: Data Pre-processing

4 Illustration and Figure

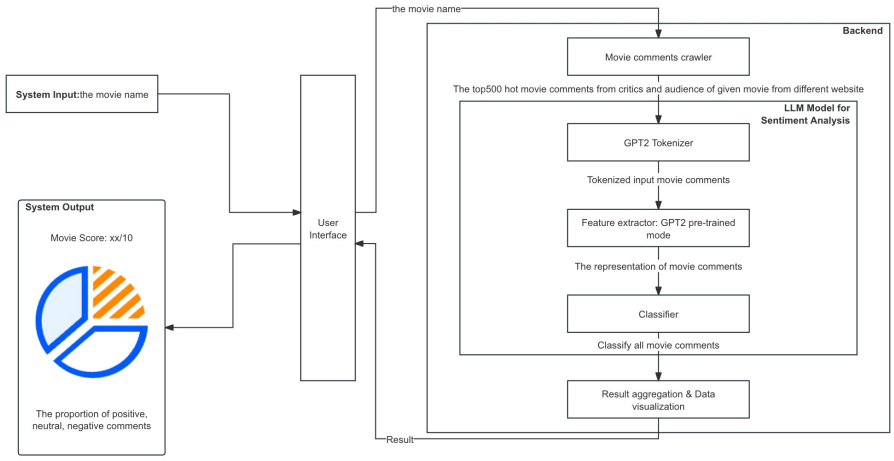

Figure 7: Software Architecture

Application Architecture Our application(Figure 7 begins with the user inputting the name of a movie in the frontend website. This input triggers the backend processes, starting with a movie comments crawler.

The crawler's job is to collect the top 500 hot movie comments from various websites including Rotten Tomato and IMDB, including 200 critics' and 300 audience opinions. These collected comments are processed by a LLM, specifically designed for sentiment analysis one by one, including tokenization, feature extraction using a GPT-2 pre-trained model.

The resulting representation is then classified into labels as positive, negative and neutral sentiments by the classifier. And these results are aggregated and

passed to a weighted scoring system(Figure 8).

Finally, the frontend will display the weighted score, the proportion of each label in two groups of people.

Sentiment Label Conversion Formulas The sentiment label conversion formulas are:

Suppose we have:

1. $\text{Positive}_{\text{audiece}}$, $\text{Negative}_{\text{audiece}}$, $\text{Neutral}_{\text{audiece}}$, $\text{Positive}_{\text{critic}}$, $\text{Negative}_{\text{critic}}$, $\text{Neutral}_{\text{critic}}$ are the number of comments of different labels in different groups.
2. $\text{Weight}_{\text{audience}}$, $\text{Weight}_{\text{critic}}$ are comment weight of different groups, and $\text{Weight}_{\text{critic}} = 2 \cdot \text{Weight}_{\text{audience}}$
3. $\text{Score}_{\text{positive}} = 3$, $\text{Score}_{\text{neutral}} = 2$, $\text{Score}_{\text{negative}} = 1$

Raw score of a movie is:

$$\text{Score}_{\text{raw}} = (\text{Positive}_{\text{audiece}} \cdot \text{Score}_{\text{positive}} + \text{Negative}_{\text{audiece}} \cdot \text{Score}_{\text{negative}} + \text{Neutral}_{\text{audiece}} \cdot \text{Score}_{\text{neutral}}) \cdot \text{Weight}_{\text{audience}} + (\text{Positive}_{\text{critic}} \cdot \text{Score}_{\text{positive}} + \text{Negative}_{\text{critic}} \cdot \text{Score}_{\text{negative}} + \text{Neutral}_{\text{critic}} \cdot \text{Score}_{\text{neutral}}) \cdot \text{Weight}_{\text{critic}}$$

To normalize the score to 10:

$$\text{Score}_{\text{normalized}} = \frac{Score_{raw}}{MaxPossibleScore} \cdot 10 \text{ , where MaxPossible} = (300 \cdot \text{Score}_{\text{positive}} \cdot \text{Weight}_{\text{audience}} + 200 \cdot \text{Score}_{\text{positive}} \cdot \text{Weight}_{\text{critic}}$$

Figure 8: Label Conversion Formulas

5 Architecture and Software

5.1 Architecture of Model

The design of our model will accept the movie reviews as inputs and predict the sentiment label(negative, neutral, positive) of those reviews. First, a input view will be convert a to a fixed length vector(1x128) and then the feature extractor, which is the GPT-2-medium pre-train model from hugging and consists of 24 layers of decoder-only Transformer blocks, each with 16 attention heads and a hidden size of 1024, will embedded this 1x128 vector into a 128x1024 hidden states. Subsequently, a pooling operation is employed, reducing the dimensions of hidden states from 128 to 1 by taking the only last hidden state. This pooling output is the representation vector of the input review(1x1024). Finally, the classifier(refer to Figure 9) will predict the probability distribution over the three classes of the input review based on the above representation vector.

5.2 Training and Inference Mode

Our model work slightly different in training mode and inference.

Training mode For the training mode(refer to Figure 10), after the classifier generates the probability distribution of the sentiments, we quantifies the difference between predicted probability distribution and the actual label by the

```
(classifier): Classifier(
  (sigmod): Sigmoid()
  (softmax): Softmax(dim=-1)
  (model): Sequential(
    (0): Linear(in_features=1024, out_features=32, bias=True)
    (1): Dropout(p=0.1, inplace=False)
    (2): ReLU()
    (3): Linear(in_features=32, out_features=3, bias=True)
  )
```

Figure 9: Classifier architecture. We add a dropout layer to the classifier model to reduce overfitting

cross entropy loss. We then use this loss to adjust the model parameters through backpropagation. It is important to note that during this training phase, we update not only the parameters of the classifier but also those of the feature extractor. This is because training the feature extractor can help our model learn a representation vector that is more relevant to our specific task.

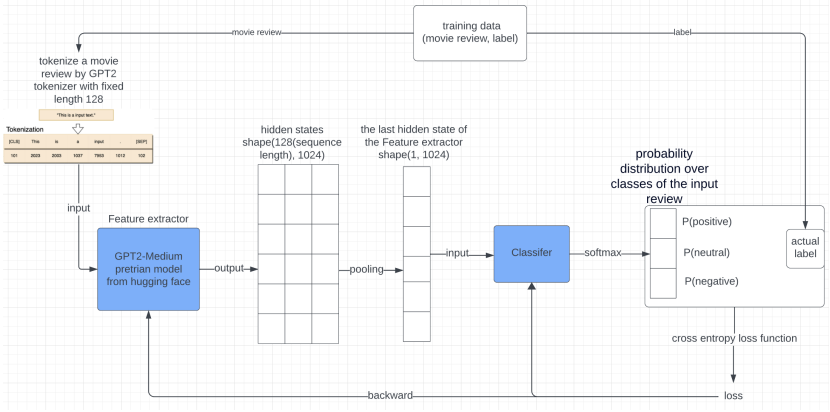

Figure 10: Training mode workflow

Inference mode Different from training mode, our classifier will greedily pick the class with the highest probability from the generated probability distribution as the predicted sentiment label for the input review(refer to Figure 11).

6 Baseline Model

6.1 Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) is a popular and powerful machine learning algorithm that belongs to the class of ensemble learning methods. Unlike a decision tree, which uses a single estimator to make predictions, XGBoost employs several estimators. Each estimator can learn from the errors of the

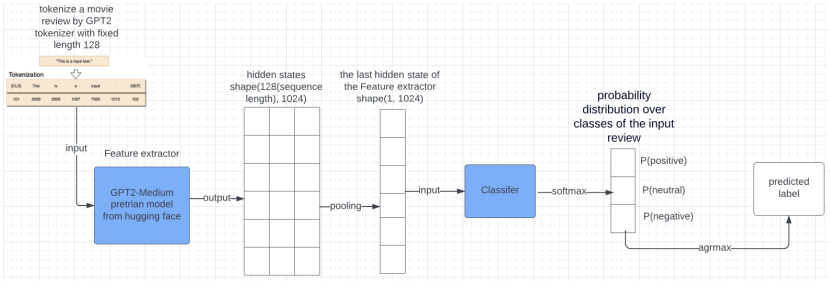

Figure 11: Inference mode workflow

previous one by assigning higher weights to the misclassified instances during training. This not only improves its performance but also reduces overfitting [3].

6.2 Term Frequency-Inverse Document Frequency

Term Frequency-Inverse Document Frequency(TD-IDF) vectorizer, a count-based vectorizer, which converts a collection of raw text into a matrix by calculating the importance for each term(TD-IDF score).

6.3 Baseline Model Architecture

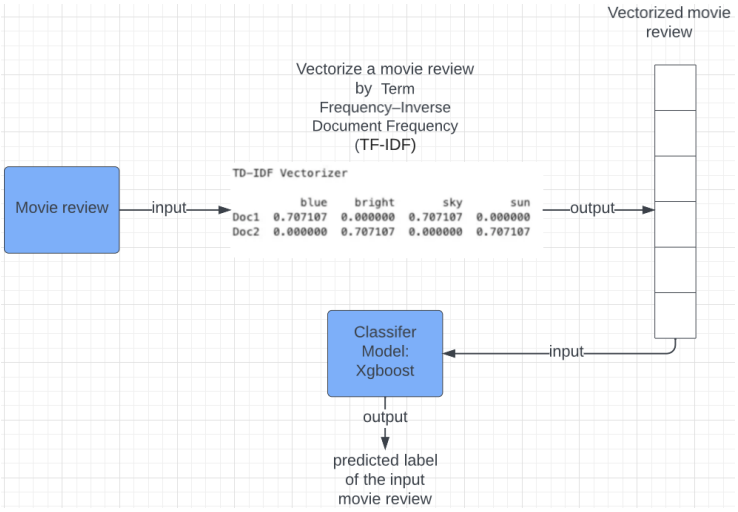

Figure 12: Baseline Model Architecture

Our baseline model supports the same functions as our model, which works on predicting the sentiment label of the input movie reviews. The baseline model first embeds a raw movie review into a numerical vector by the TD-IDF

Vectorizer, and then the XGBoost classifier predicts the sentiment label of the input movie review based on the embedded vector(refer to Figure 12).

7 Quantitative Results and comparison

$$\text{precision of class i} = \frac{\text{the number of class i movie reviews are classified as class i}}{\text{the number of movie reviews are classified as class i}} = \tag{1}$$

$$\text{recall of class i} = \frac{\text{the number of class i movie reviews are classified as class i}}{\text{the number of class i movie reviews}} \tag{2}$$

$$\text{recall of class i} = \frac{\text{confusion matrix(i,i)}}{\text{sum of row i of confusion matrix}} \tag{3}$$

$$\text{F1 class i} = 2 \times \frac{\text{precision of class i} \times \text{recall of class i}}{\text{precision of class i} + \text{recall of class i}} \tag{4}$$

To evaluate the performance of our model, we use accuracy, which means the percentage of movie reviews for which the sentiment is correctly classified by the model. However, recognizing the imbalanced nature of the data, relying solely on the accuracy metric is inadequate for evaluating model performance. Therefore, the inclusion of the marco-F1 score (average f1-score of F1 score for each sentiment class, measuring the model performance on distinguish the label on each sentiment class) and confusion matrix(matrix(i,j) represents the number of class i movie reviews are classified as class j. In our matrix, class 0 represents negative sentiment, class 1 represents neutral sentiment, class 2 represents positive sentiment) becomes essential. Finally, the recall score will be applied to assess the percentage of movie comments in each sentiment class that are successfully identified.

7.1 Performance of Project Model

According to the Figure 13, Our project model correctly predicts the sentiments of 92% of movie comments and achieves a 0.87 Macro F1 score on the test dataset, indicating overall good performance. In terms of class-specific generalization, the model excels in distinguishing negative and positive comments, successfully identifying 93% of negative comments and 97% of positive comments. However, it shows average performance on neutral comments, identifying only 69% of them(see Figure 15). Besides, the confusion matrix(Figure 14) shows that 407 out of 1751 neutral comments are misclassified as positive comments, which reveals that the model tends to misclassify neutral comments as positive.

7.2 Comparison between Baseline model and Project Model

According to the Figure15, given the same dataset, the baseline model achieve a slightly lower level of accuracy(0.85) and marco-f1 score to the project model.

Furthermore, there is an obvious gap of 0.2 in the recall score for the neutral class between the baseline model and the project model, suggesting that the GPT2 based model has superior performance in text sentiment feature extraction.

7.3 Effectiveness of Expanding dataset

In Section 3.1, we introduce a novel approach to overcome the challenges associated with poor model generalization to neutral movie comments, which result from a scarcity of class neutral comments. This is achieved by leveraging comments generated by GPT4. This section is dedicated to validating the effectiveness of this proposed method.

According to the Figure 16, the inclusion of the generated comments has a minor impact on accuracy and macro-F1 score. However, it significantly boosts the recall of the neutral sentiment class, increasing it from 0.39 to 0.69. This suggests that expanding the dataset by generating comments can mitigate data imbalance and enhance the model's generalization ability for comments with neutral sentiment. This result confirms the effectiveness of our approach.

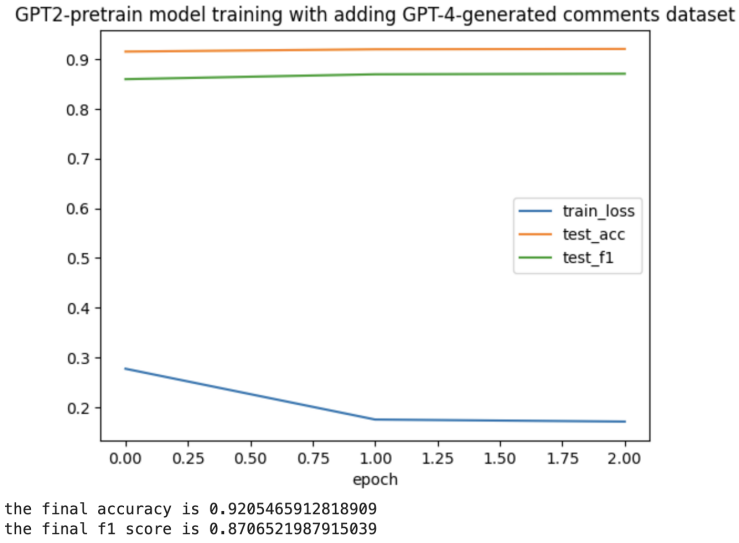

Figure 13: GPT2 based model training curve

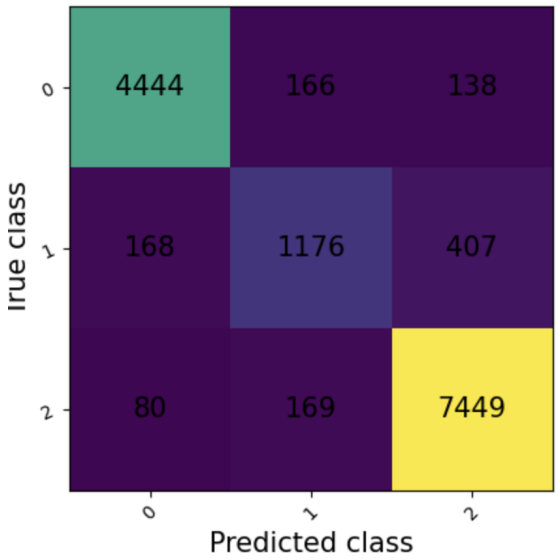

Figure 14: GPT2 based model confusion matrix

Model	Marco F1 score	accuracy	R1 score for negative sentiment	R1 score for neutral sentiment	R1 score for positive sentiment
GPT-2 pre-train model(Project Model)	0.87	0.92	0.93	0.69	0.97
Baseline Model	0.78	0.85	0.83	0.49	0.95

Figure 15: Comparing Performance of GPT2-Based model(project model) and Baseline Model

Model	Marco F1 score	accuracy	R1 score for negative sentiment	R1 score for neutral sentiment	R1 score for positive sentiment
GPT-2 pre-train model train with generated comments dataset (Project Model)	0.87	0.92	0.93	0.69	0.97
GPT-2 pre-train model train without generated comments dataset	0.75	0.87	0.91	0.39	0.954

Figure 16: Comparing Performance Before and After the Inclusion of a GPT4-Generated Neutral Comments Dataset

8 Qualitative Results

The qualitative result example of our application is shown in the Figure 17, Figure 18. We have created a front-end website which allows users to input a film name they want to analyze, and the application will provide them with our comment sentiment proportions and weighted score based on the sentiment analysis.

It's obvious our weighted average score is better than the score in Rotten Tomatoes and IMDb (Figure 19). Since we can see that there is a big gap between

the score from audiences and critics, but our weighted score can mitigate such biases by providing a more balanced score that accounts for the intensity of sentiment by assigning greater weight to critic reviews. Because critics often have more experience and expertise in film evaluation.

However, for the latest movies, the limitation of the number of reviews will lead to the poor effect of the scoring system.

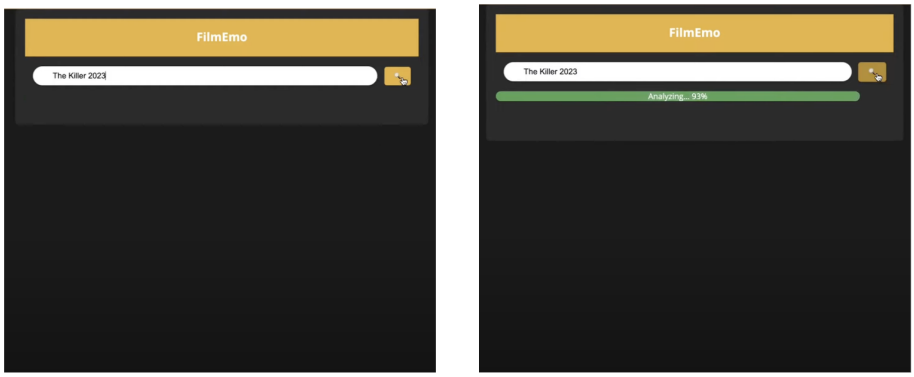

Figure 17: Main Page and Analyze A Film

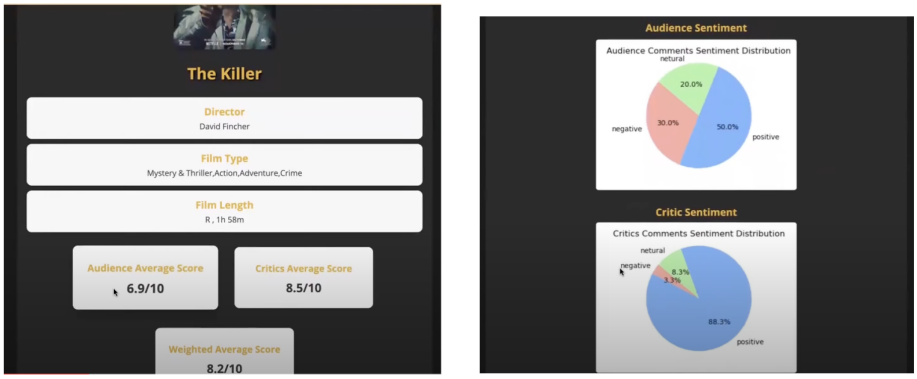

Figure 18: Display Weighted Scores and Sentiment Proportions

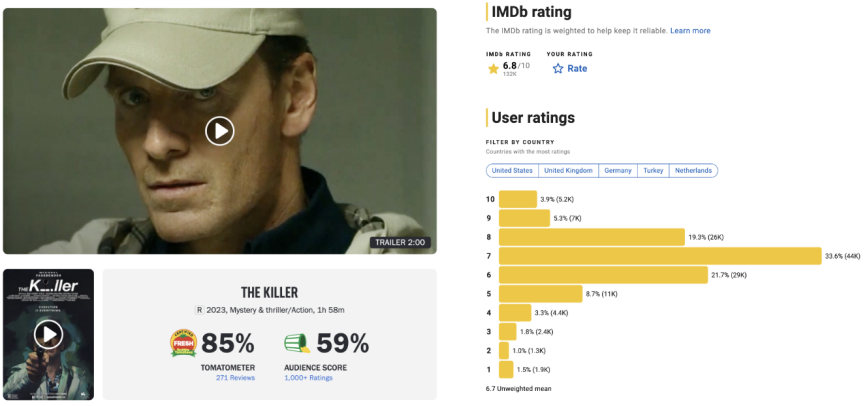

Figure 19: Rotten Tomatoes and IMDb Scores

9 Discussion and Learning

Based on the training curve, we observe a successful reduction in training loss concurrent with increases in test F1 scores and accuracy. Besides, the recall of neutral class movie reviews has significant improvement after addressing the data imbalance issue. This suggests that our system is performing effectively. However, our model is still struggling on distinguish neutral comments and positive comments. We attribute this challenge to our labeling approach, which solely relies on ratings and may not consistently reflect the sentiment expressed. For example, positive comments might occasionally be associated with neutral ratings. Inspired by the idea of balancing the dataset with GPT-4 generated comments, we are keen to investigate in exploring the feasibility of leveraging GPT-4 to label the dataset by providing minimal manual labeling examples. Furthermore, we are interested in exploring the potential of distilling the knowledge from a very large model to a relative small model by supervised training the small model on the dataset labeled by the large model.

10 Individual Contributions

Work	Contributor
Data collection	Jianning Qu
Web crawler	Jianning Qu
Data pre-processing	Jianning Qu
Build Baseline Model	Jianning Qu
Weighted Score Formulas	Jianning Qu
Result aggregation and Data visualization	Jianning Qu
Build frontend and backend application	Jianning Qu
Using GPT4 to generate movie comments data	Jiaxi Lyu
Data pre-processing for Project Model	Jiaxi Lyu
Design and build Project Model	Jiaxi Lyu
Experimented with various pre-train GPT-2 models to improve model performance	Jiaxi Lyu
Training Project Model and Fine-tuned hyper-parameters	Jiaxi Lyu
Construct output plots	Jiaxi Lyu

Table 1: Individual Contributions

References

[1] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2018.

[2] L. N, "Imdb dataset," 2020, accessed: 2023-11-01. [Online]. Available: https://www.kaggle.com/code/lakshmi25npathi/sentiment-analysis-of-imdb-movie-reviews

[3] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco, CA, USA: ACM, 2016, pp. 785–794.

11 Permission

Jianniing Qu permission to post video: yes
Jianniing Qu permission to post final report: yes
Jianniing Qu permission to post source code: yes

Jiaxi Lyu permission to post video: yes
Jiaxi Lyu permission to post final report: yes
Jiaxi Lyu permission to post source code: yes