**Sentiment Analysis of popular sequels based on Reddit comments: attempting to understand the performance of sequels over time**

**Ewald Seelich, Maelys Lafaurie, Phuong Anh, Sora Schultz, Lillian Hsiao**

**Abstract**

In the past years, we can observe a decline in sequel performance, this motivated us to attempt to understand audience sentiment dynamics regarding famous sequels movies in the past. Are there any clear trends we can observe that positive sentiments decline through time? In order to achieve this, we collected Reddit comments via a Reddit API. We decided to use two famous series with the same number of sequels: Pirates of the Caribbean, and Twilight. They were selected for their known popularity, and critics. The collected data was preprocessed and then classified using two different approaches: VADER as a baseline and our own machine learning model.

In our results, we found an accuracy of 0.546 for our own developed model. Although our accuracy is low, a similarly low accuracy of 0.507 from the VADER model (which is generally a reliable model for sentiment analysis), leads us to believe that low accuracy is not a flaw in our code or model. It is an issue concerning the complexity of our extracted comments and labeling. Moreover, the VADER model consistently displays a higher sentiment score than the BERT model. However, BERT model’s predicted sentiment scores are more likely to match the average audience and critics’ rating scores than VADER model. Overall, the Pirates of the Caribbean sequel often holds a higher sentiment score than Twilight, which reflects the higher level of audience engagement.

**Introduction**

In recent years, we noticed how many movies were coming up with sequels, but also how they haven’t been doing as well. Indeed, the concept of sequels was very popular in the past two decades. We can observe very popular movies that have now become a part of our culture. Having grown up with these movies and sequels, we decided it would be interesting to investigate, and attempt to understand how sequels are viewed today, and can we see a clear decline in positive sentiments with time. While brainstorming ideas, we realized that there are very diverse movie sequels, and decided to focus on two: Pirates of the Caribbean and Twilight. The reason we chose those two is that they both had a total of five movies, and are very popular. This ensured we would gather sufficient data for our analysis. Additionally, we took the decision to only select two movie series to accentuate our comparison analysis. The idea was then to find a public API in order to extract comments, and sentiments based on each movie. We decided to use Reddit comments because it would ensure a sufficient amount of comments, and it is easily accessible due to their open API.

Research Question: How have sentiments towards the Pirates of the Caribbean and Twilight movies changed over time, and does it have any correlation with box office overall ratings?

Overall, our main goal is to understand people’s sentiments towards these series, and to see if the sentiments evolve in a certain way with time. We want to be able to see if we can observe a trend based on our analysis that could help make future predictions of how well a next sequel would perform. We decided to compare them with each other to attempt to extract certain patterns. However, we are also aware that sentiments towards a movie can be very personal. Our conclusions might be that it is, in fact, complicated to predict any trends of effects based on Reddit extracted comments for movie series sentiment analysis.

**Related Work**

***Understanding Sentiment Analysis***

Sentiment analysis can be approached in two main ways: lexicon-based tools or machine learning models (Samuel & Marikannan, 2020). Traditional lexicon-based sentiment tools rely on large dictionaries—such as WordStat or Bill McDonald's 2014 Master Dictionary—which contain thousands of words each labeled as positive, neutral, or negative. These individual word sentiments are then aggregated to determine the overall sentiment polarity of a text. Different lexicon tools may also vary in how they classify words (Hossen & Dev, 2021). While these tools are simple and interpretable, they lack nuance and the ability to factor in context. However, this can be improved with more complex adaptations, such as flipping the sentiment when the word “not” is in front of another word (e.g. “not good” would be a negative sentiment). Hossen and Dev recommend more advanced models that can account for such context-based shifts in meaning (Hossen & Dev, 2021).

The second approach to sentiment analysis involves machine learning, which requires labeled training data. Common models in this category include Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Naive Bayes, and Random Forest. The accuracy of these models often depends on the context in which they’re applied. For instance, one sentiment analysis study on movie reviews using Twitter’s API found that SVM was the most effective model for predicting movie sentiment and ratings (Samuel & Marikannan, 2020). In contrast, another study concluded that Logistic Regression was the most accurate model for the same task of movie analysis (Teja, Sai, Kumar, & Manikandan, 2018).

Interestingly, a comparative study of lexicon-based and machine learning approaches found no significant difference in their accuracy scores—contrary to much of the previous literature (Dhaoui, Webster, & Tan, 2017). Instead, the authors recommended a hybrid approach. This inspired our decision to evaluate both lexicon-based and machine learning techniques in our research. We decided to use a lexicon-based model as our baseline *and* develop a machine learning model to analyze both comparatively for a more nuanced and robust sentiment classification.

***Choosing a Baseline Model: Vader Research***

We use VADER as our lexicon-based baseline model based on existing research and examples. VADER stands for Valence Aware Dictionary for sentiment Reasoning, and it is a rule and lexicon based sentiment analysis tool that produces a sentiment score of positive, neutral, negative, or compound represented from -1 to 1 (Pano & Kashef, 2020). It can sufficiently deal with stopwords, slang, emojis, emoticons, etc. and is word-based, meaning it does not understand context. However, it has been shown to be consistent in different usages and especially strong in sentiment analysis of “microlog-like contexts” (Hernández-Pérez et al., 2024). Because Reddit comments are often lengthy, we decided to use VADER as the baseline model that would produce most accurate results without training and labeling.

***Using Deep Learning Model BERT for sentiment analysis***

For our multi-modal machine learning approach, we decided to incorporate BERT, which stands for Bidirectional Encoder Representation from Transformers, and is designed to pretrain deep bidirectional representation from unlabeled text. Pre-trained BERT models can be fine-tuned with only an additional output layer (Chang et al., 2018). Moreover, advancements in transformer-based pre-trained models show that they have received better results in the text classification task. Following the launching of the BERT model, further research explores the features of BERT variants with three BERT-based models, including RoBERTa, ALBERT, and distilBERT. Specifically, the compressed BERT model (distilBERT) is discovered to establish concrete results for sentiment analysis. “The distilBERT model retains 97% of the language understanding capabilities and the size is reduced by 40% in contrast to BERT based model.” (Batra et al., 2021). The prevailing utility of the BERT-based model exhibits its capability for contextual-based sentiment analysis. Therefore, our present research work aims to show the nuance and accuracy of a deep-learning incorporated model in sentiment analysis.

**Data Description**

1. Data Collection:

The goal was to collect Reddit comments based on the different movies of each sequel in order to gather sufficient data for our sentiment analysis. We obtained Reddit API Credentials, in order to scrap the necessary data. For the first try, we would just input the name of the movie, and find the Subreddit comments associated. However, we found that using this technique provided too many unrelated data. As a solution, we selected more precise Subreddits to enable more precise scrapping. Hence, our code would loop through each Subreddit in your list. For each, run a relevance‐sorted, all‐time search limited to previously fixed amounts of posts. Our code then expanded nested comments fully, then iterated through every comment in each thread. For each comment, we recorded the Subreddit name, post ID and title, comment ID, timestamp, body text, and author. This was repeated for each movie, giving us final results of ten distinct datasets. However, given the data was extracted from Reddit, we needed to do quality checks to remove duplicates, filter out deleted or bot comments.

In order to navigate the correlation between two sequels’ sentiment score and box office gross value or movies’ rating, we collect the cleaned database of box office gross and rotten tomatoes’ film ratings. We later sort the movies in the database in order to collect its gross and average scores from critics and audience.

1. Data preprocessing

In this step, we focus on preparing and preprocessing the text so that the raw text data can be input to BERT models. We perform text cleaning in order to remove URL links, symbols, stop words or common words. The output of the cleaning step is saved in the *cleaned\_body* column. *label* has the label of input features. In our case, the *label* implies the sentiment polarity of each movie. After completely loading the labeled data, we split the dataset into a train set and a test set. The dataset is then oversampled in order to address potential imbalances between the number of negative, neutral and positive comments. During the data transformation step, the text comments are tokenized, trained, and tuned in order to fit the classic model training process. The preprocessing step ensures the removal of unnecessary and irrelevant information, segregation of text to single tokens and transformation to adapt further machine learning training.

**Methodology**

The creation of our model evolved from a first very basic machine learning model, into our final usable model. Here is a detailed explanation of the different steps, and models we used before obtaining one we were satisfied with.

For easier loops and a more structured overview of the movies, numbers instead of their names were used for the following steps. The movies and corresponding numbers are shown in [Table 1](#opdlhwhbnb7f).

Table 1 Move Name and Corresponding Number

| **Movie Name** | **Number** |
| --- | --- |
| Pirates1 - Curse of Black Pearl | 1 |
| Pirates 2 - Dead Man's Chest | 2 |
| Pirates 3 - At World's End | 3 |
| Pirates 4 - On Stranger Tides | 4 |
| Pirates 5 - Dead Men Tell no Tales | 5 |
| Twilight 1 | 6 |
| Twilight - New Moon | 7 |
| Twilight - Eclipse | 8 |
| Twilight - BD 1 | 9 |
| Twilight - BD 2 | 10 |

1. **Pre-trained VADER Sentiment Analysis Model**

Following our first phase of data collection and cleansing, we started with the initialization of the VADER model. The idea behind this is to create a benchmark that we can test against our independently created model. The data set from Twilight 1 is provided here as an example. As individual data impurities occurred during manual labeling of the data, it was not possible to perform a loop over all other data sets. Instead, the Vader model was initially performed separately for all films. Therefore, manual work was required to go through each of the 10 datasets.

*import pandas as pd*

*twilight\_labeled = pd.read\_excel('6\_labeled.xlsx')*

Essential cleanups that still needed to be done include removal of hidden “spaces”, spelling errors in labeling, and missing values in labeling:

*twilight\_labeled = twilight\_labeled.rename(columns={"sentiment ": "sentiment"})*

*twilight\_labeled = twilight\_labeled.dropna(subset=['sentiment', 'comment\_body'])*

*twilight\_labeled['sentiment'] = twilight\_labeled['sentiment'].replace({'positive ': "positive", 'negative ': "negative", 'neutral ': "neutral"}*

Next, the initialization phase of the VADER algorithm begins. This is triggered by importing the “SentimentIntensityAnalyzer” from the “vaderSentiment.vaderSentiment” library. Variables are then defined, which assign a “compound” score and, based on this, a sentiment

*from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer*

*analyzer = SentimentIntensityAnalyzer()*

*def get\_vader\_sentiment(text):*

*scores = analyzer.polarity\_scores(text)*

*compound = scores["compound"]*

*return "positive" if compound >= 0.05 else "negative" if compound <= -0.05 else "neutral"*

VADER uses a compound score to determine sentiment, with thresholds of 0.05 and -0.05 to classify sentiment as positive, neutral, or negative, respectively. A compound score represents the overall sentiment of a piece of text based on its lexical features (words and phrases) and other factors like capitalization and punctuation. Specifically, a compound score of 0.05 or higher indicates positive sentiment, a score between -0.05 and 0.05 indicates neutral sentiment, and a score of -0.05 or lower indicates negative sentiment.

Once Vader has been initialized and the necessary variables have been determined, the algorithm is applied to the data set and the results of the classification are recorded in separate columns. Afterwards, the results are saved as a csv file and exported for further processing.

*Twilight\_labeled["vader\_sentiment"] = twilight\_labeled["comment\_body"].astype(str).apply(get\_vader\_sentiment)*

*twilight\_labeled.to\_excel('6\_labeled\_vader.xlsx', index=False)*

However, we are not only interested in the final results of Vader, but our goal is to obtain important metrics. Therefore, we also calculated the Accuracy Score, the Macro Average and the Weighted Average, which are provided by the Confusion Matrix. This analysis is made possible by the fact that we as a group have already manually labeled a subset of the data sets (per movie). In this case, the manually performed classifications are our “true sentiments”

The relevant libraries for the evaluation are fed in accordingly

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

import seaborn as sns

import matplotlib.pyplot as plt

The evaluation itself is then carried out by the variable “accuracy”, modified from the variable “accuracy\_score” already contained in sklearn.metrics

*valid\_data = twilight\_labeled.dropna(subset=['vader\_sentiment'])*

*accuracy = accuracy\_score(valid\_data['sentiment'], valid\_data['vader\_sentiment'])*

*print(f"VADER Accuracy: {accuracy:.4f}")*

*print("\nVADER Classification Report:")*

*print(classification\_report(valid\_data['sentiment'], valid\_data['vader\_sentiment']))*

Finally, the graphic is initialized and extracted. The aim is to carry out a graphical comparison with VADER as the baseline model and our self-made model

*cm\_vader = confusion\_matrix(valid\_data['sentiment'], valid\_data['sentiment'], labels=['positive', 'neutral', 'negative'])*

*plt.figure(figsize=(6, 4))*

sns.heatmap(cm\_vader, annot=True, fmt='d', cmap='Blues', xticklabels=['positive', 'neutral', 'negative'], yticklabels=['positive', 'neutral', 'negative'])

plt.title('VADER Confusion Matrix: Twilight 1')

plt.xlabel('VADER Sentiment')

plt.ylabel('True Sentiment')

plt.savefig('twilight1\_vader\_confusion\_matrix.png')

plt.show()

1. **DIY Machine Learning Model**

Once our baseline model VADER was in place, we started with the creation of our own sentiment analysis model. The first step in the creation of our model was manual labelling for every movie. We decided to label 300 randomly selected comments from each movie dataset. The very first creation is the code: 1-Basic\_ML\_model.py. With this one, we loaded a small, manually labeled sample and performed train/validation split and class balancing via under/oversampling. We vectorized our text with TF-IDF (1–3 grams, top 5 000 features) and trained a calibrated LinearSVC for probabilistic outputs. Then, evaluated our model with accuracy, classification reports, and confusion matrix. However, we were not satisfied with the results, hoping to reach higher accuracies with more advanced models.

As a more sophisticated model, we use the DistillBERT deep learning model. The second test is based on the code: 2-finetune\_sentiment, which was fine tuned and evolved into 3-FineTunedDistilBERT. For both scripts, we loaded manually labeled files and mapped the three sentiment classes to integers: negative:0, neutral:1, positive:2. We then performed a stratified 80/20 train test split to preserve class proportions. However, in our second fine tuned version, we coded an explicit oversampling step that balances the training set by duplicating minority-class examples up to the majority class size, improving our model’s robustness on underrepresented labels. Both scripts define a SentimentDataset subclass of `torch.utils.data.Dataset` that takes raw texts and labels, uses AutoTokenizer and returns token-IDs and attention masks along with the integer label tensor. When it comes to the training arguments, the first script uses three epochs, which is the default learning rate. However, our fine-tuned model uses five epochs for more training, a learning rate of 2e-5 and a weight decay of 0.01 to regularize our data. This evolution reflects hyperparameter tuning and best practices for transformer fine‐tuning. Once the training and evaluation step has been completed, our fine-tuned model adds a threshold grid search over negative and positive softmax probabilities to optimize macro-F1 on the held-out set. This targeted tuning helps recover the often-underpredicted “negative” class, yielding a more balanced classifier. Overall, our fine-tuned model addresses class imbalance issues, evolving the hyperparameters and evaluation frequency. Together, these refinements yielded a more robust, balanced, and efficient DistilBERT fine‐tuning and inference workflow compared to the initial code. However, we were still not satisfied with the results, as accuracy was barely over 40%.

Our final model consists of three scripts that run all three models (DistilBERT, SVC, LR) and achieves an ensemble weight search. To begin with, sentiment\_utils. is a utility library that encapsulates all of your data cleaning, model training, threshold tuning, and ensemble logic. The key functions present are:

* clean\_text(text: str) which removes URLs, HTML tags, markdown, mentions, punctuation, numbers, lowercases and removes extra whitespace and returns a normalized string ready for tokenization and vectorization
* load\_labeled\_data(labeled\_map: Dict[str,str]) which reads each <movie\_id>\_labeled.xlsx into a DataFrame and applies clean\_text to produce a cleaned\_body column. This step concatenates them, maps sentiment strings to integer labels negative:0, neutral:1, positive:2
* split\_and\_oversample(df: DataFrame) → (train\_df, test\_df) which stratified 80/20 split on the integer labels. This returns a balanced train\_df and an untouched test\_df
* train\_transformer(train\_df, test\_df) → (trainer, tokenizer, model) which wraps data in a custom SentimentDataset (tokenized with AutoTokenizer). This step also fine-tunes distilbert base uncased for 3–5 epochs with Trainer/TrainingArguments and monitors f1\_macro on test\_df and saves the best checkpoint.
* tune\_negative\_threshold(trainer) → float which runs trainer.predict() on test\_df to get softmax probabilities and returns the optimal neg\_thr
* train\_linear\_svc(train\_df, test\_df) → (vectorizer, svc\_clf) This step is the TF-IDF vectorization (1–3-grams, top 5 000 features) on train\_df.cleaned\_body and wraps LinearSVC in CalibratedClassifierCV (5-fold) for probability estimates. It then Fits on the balanced train\_df and returns both the fitted vectorizer and svc\_clf
* train\_logistic\_regression(train\_df, test\_df) → (vectorizer, lr\_clf). This step uses the same pipeline as the SVC but with LogisticRegression and returns its own TF-IDF vectorizer and classifier
* ensemble\_predict(models, vectorizers, texts, neg\_thr, weights) → List[int]. In this step, each text gets: transformer logits → softmax → probs\_tf, and returns the list of integer predictions

Once this script was done, we were able to implement it in our main model script: Final-model.

This is the script with the full train test pipeline on our manually labeled data and produces different evaluation artifacts. Such as argument parsing, data loading and splitting where we

* Builded labeled\_map = {m: f'{m}\_labeled.xlsx'}
* Loaded and cleaned all labeled data via load\_labeled\_data()
* Splitted and balanced into train\_df / test\_df via split\_and\_oversample()

Then we proceeded with model training and tuning with these different elements:

* Transformer: train\_transformer() → returns (trainer, tokenizer, model)
* Threshold: tune\_negative\_threshold(trainer) → neg\_thr
* Classical: train\_linear\_svc() & train\_logistic\_regression()

Then we proceeded with ensemble weight search with nested loops over plausible w\_tf, w\_svc, w\_lr combinations which calls ensemble\_predict(...) on test\_df.cleaned\_body and picks the weight triple that yields highest accuracy. The final step of this is evaluation and visualization where we print overall accuracy, classification report, and confusion matrix which will be presented in the results section. Finally, when we were satisfied with our accuracy results, we re-ran the entire training pipeline and then applied the final ensemble to every comment in our full datasets which labelled all our data. Once we had all our labelled data, we proceeded with our analysis and visualization.

To add more information, we used the BoxOfficeAndBudget.ipynb data set, in which we ingested box-office revenue and production budget figures alongside sentiment outputs. Then, visualized relationships (scatter plots, correlation coefficients) between average sentiment and movie performance.

The final step of our method was creating a success prediction code. The script loads our historical CSV (with columns series, mean\_sentiment (derived from the mean values provided by the BERT model), budget\_musd, rating\_pct, box\_office\_musd), drops any rows with missing values, encodes “Pirates” vs. “Twilight” as a binary feature, then trains a RandomForestRegressor. It performs 5-fold cross-validation to report an average R² score and finally fits on the full dataset. You can then pass in new feature triples for a proposed Pirates and/or Twilight film, and it will output the predicted global box-office (in USD) and tell you which franchise is expected to earn more. An R² far below zero means our model’s squared errors exceed the variance of actual box-office values, which on just 10 data points with huge revenue swings and only four simple predictors is to be expected. With so few movies, high target variance (hundreds of millions USD), and missing real drivers like marketing spend or release timing, cross-validation becomes extremely noisy. For demonstrating the end-to-end pipeline feature handling, CV routine, training, and inference, this negative R² is simply a consequence of too little data and too few features, not a flaw in the code itself.

**Results & Analysis**

Our model resulted in an accuracy level of approximately 0.546. Below is its precision, recall, and f1-score according to classification metrics ([Table 2](#bojutmduoiij)). This was based on training the model with all 10 of our labeled dataset samples. We also tried training the model on just one franchise movie respectively to see if there would be unique patterns to a movie that improved training accuracy. However, our accuracy results proved around the same (only Pirates of the Caribbean training data: 0.597 and only Twilight training data: 0.539).

Table 2: Our Model Classification Metrics by Class

| **Class** | **Precision** | **Recall** | **F1-score** | **Support** |
| --- | --- | --- | --- | --- |
| Negative | 0.46 | 0.51 | 0.48 | 141 |
| Neutral | 0.56 | 0.59 | 0.58 | 182 |
| Positive | 0.61 | 0.53 | 0.57 | 188 |

This classification accuracy metric shows us that we have the highest precision in positive sentiments and lowest precision in negative, while our overall best-performing class is the neutral one. We think this could be because negative sentiments are harder to catch with subtle negativity such as sarcasm or complex criticisms.

Table 3: Our Model Correlation Matrix

|  | **Pred: Negative** | **Pred: Neutral** | **Pred: Positive** |
| --- | --- | --- | --- |
| **Actual Neg** | 72 | 42 | 27 |
| **Actual Neu** | 38 | 108 | 36 |
| **Actual Pos** | 46 | 43 | 99 |

Again, our correlation matrix ([Table 3](#qv3c7h39a0c5)) shows that our model does moderately alright with positive and neutral sentiments, but struggles with accuracy for negative sentiments. Although this is lower accuracy than we would have preferred, our model is actually doing better than our baseline, VADER.

VADER has an accuracy of approximately 0.507. Below is the classification accuracy metric and correlation matrix for our baseline model ([Table 4](#ei9fo49t2tmi)).

Table 4: Vader Classification Metrics by Class

| **Sentiment** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| Negative | 0.43 | 0.44 | 0.44 | 768 |
| Neutral | 0.68 | 0.33 | 0.44 | 1043 |
| Positive | 0.49 | 0.73 | 0.59 | 1064 |

Table 5: Vader Correlation Matrix

|  | **Predicted Positive** | **Predicted Neutral** | **Predicted Negative** |
| --- | --- | --- | --- |
| **Actual Positive** | 778 | 89 | 197 |
| **Actual Neutral** | 456 | 339 | 248 |
| **Actual Negative** | 354 | 73 | 341 |

Similarly to our model, VADER struggles with negative precision. In [Table 5](#ly20c17azaid), we also see a bias towards predicting positive sentiments. (810 were misclassified as positive). Additionally, we see that precision is highest for predicting neutral.

Seeing that our model’s accuracy shows similar characteristics to our baseline model and yet is slightly more accurate shows us that the lack of accuracy in our model is likely not due to a coding or machine learning flaw. It is rather because a lot of our comments are too nuanced. As we performed the manual labeling of 300 codes per movie, we realized that many comments involved multiple perspectives and multiple sentiments, so just labeling one comment as either positive, negative, or neutral may have been oversimplifying the analysis. In addition, some comments may have shown sentiments about a specific character or plot development, which again may have caused overgeneralization when we labeled that as general sentiment of the movie.

However, we still found many interesting results from comparing our model with VADER’s and looking at the movie sentiments for these two franchises over time. These results we will discuss below.

| Movie 1 | Movie 2 |
| --- | --- |
| Movie 3 | Movie 4 |

| Movie 5 |
| --- |

Figure 5.1: Sentiment analysis over time by movie release number

According to [Figure 5.1](#94ops58a8a5n), for the Twilight franchise, our model (BERT) and our baseline (VADER) seem to follow similar peaks and troughs throughout all 5 movie releases (except deviating slightly for movie 5). This is depicted by the dotted lines in the graphs. On the other hand, for the Pirates of the Caribbean franchise, our BERT, and VADER models do not align as closely to each other. If we take our model to be more accurate, this could mean that perhaps comments in Pirates of the Caribbean franchises are more complex and nuanced, which make it harder for VADER to detect the correct sentiments. It could be that the sentiments in the Twilight comments are more emotionally charged or more obviously positive, negative, or neutral, that it is easier for VADER to pick up on, and it is able to come up with an analysis closer to our model.

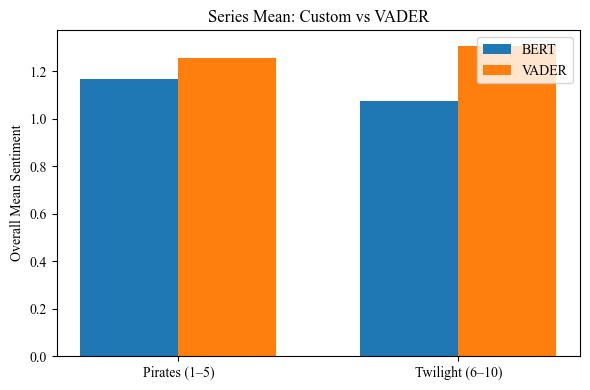


Figure 5.2: Overall Mean Sentiment Score of Two Sequels By BERT and VADER Model

According to [Figure 5.2](#89mc2p88m3r6), the overall mean sentiment scores of VADER are higher than those of BERT in both sequels. Specifically, in the BERT model, the positive sentiment score for Pirates of the Caribbean sequels is slightly higher than that of Twilight, which implies that the popularity and favor lean on Pirates of the Caribbean series. However, the VADER model shows the opposite result, with the Twilight series holding slightly higher sentiment scores than the Pirates of the Caribbean franchise.

5.3. Correlation Between Box Office’s Gross and Sentiment Score

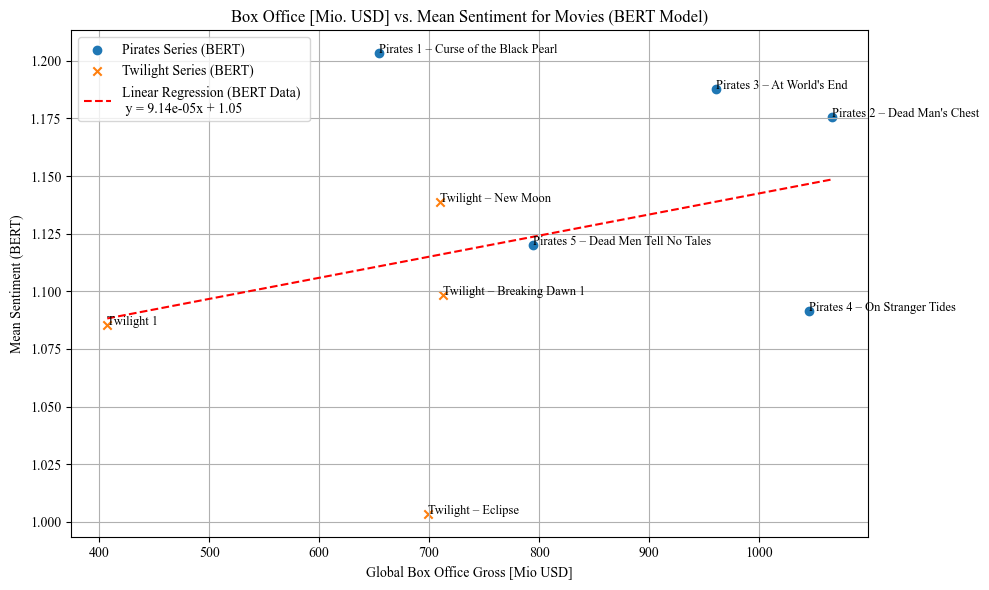


Figure 5.3: Correlation Between Global Box Office Gross and BERT Model’s Mean Sentiment Score

According to the scatter plot [Figure 5.3](#d82vhwnjuv8u), it is suggested that the higher global box office gross records, the higher average sentiment score for the movie. It is implied that movies that receive high value of gross revenue tend to have positive engagement from the audience.

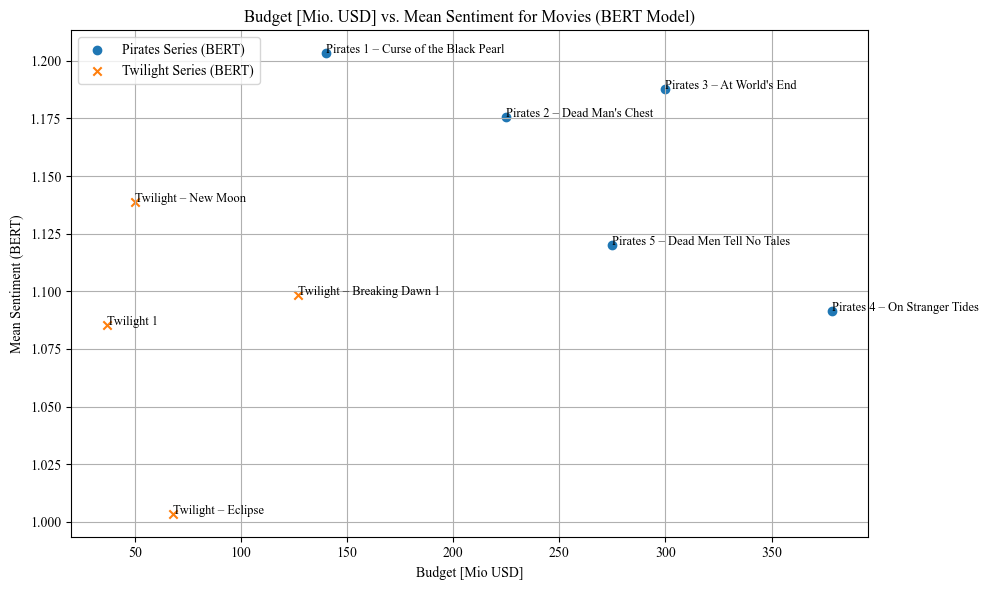


Figure 5.4: Scatter plot of Budget and Mean Sentiment Score of Both Sequels

It can be observed from [Figure 5.4](#jo9bymp67zbv) that there is a large margin in the production budget of two sequels. The Pirates of the Caribbean franchise holds a range from 145 million USD to above 350 million USD, while Twilight sequel ranges from under 50 to below 150 million USD. According to the BERT model prediction, all Pirates of the Caribbean movies have higher positive sentiment scores than the Twilight series, with the range from 1.080 to 1.2 and 1.001 to 1.130 respectively. However, even though the latter movies in Caribbean sequels received high budget investment, namely Caribbean 4 and 5, the sentiment score witnessed a decrease when compared to its predecessors.

| Figure 5.5 | Figure 5.6 |
| --- | --- |

Figure 5.5 Critics Scaled Rating and Mean Sentiment Prediction by BERT Model

Figure 5.6 Critics Scaled Rating and Mean Sentiment Prediction by VADER Model

Originally, the ratings from the Rotten Tomatoes database ranged from 0 to 100. In order to match our sentiment score, which ranges from 0 to 2, we scale the range of rating score to 0 to 2. Comparing the predicted score of both models and the average critical scaled rating of Rotten Tomatoes database from both movies, it is shown that most movies’ predicted sentiment score is higher than the scaled score from critics, except for Pirates of the Caribbean 1 movie. BERT model’s prediction on sentiment score has a lower range than that of VADER model, with the range from 1.0 to 1.2 and range from approximately 1.1 to 1.3 respectively. Moreover, BERT model’s predicted score is more likely to reach the diagonal line, where the predicted sentiment score is the same as the critics’ scaled score, than VADER model’s scores. For example, movies, namely Twilight - Eclipse, Twilight 1, Twilight - Breaking Dawn 2, and Caribbean Pirates 2, are close to the diagonal line based on BERT model, while they are further from that of VADER model.

| Figure 5.7 | Figure 5.8 |
| --- | --- |

Figure 5.7 Audience Scaled Rating and Mean Sentiment Prediction by BERT Model

Figure 5.8 Audience Scaled Rating and Mean Sentiment Prediction by VADER Model

Different from the distribution of critics scaled rating in [figure 5.5](#ai0k4hhid9l4) and [figure 5.6](#uot28wllei9), audience scaled rating and sentiment score witnessed an opposite trend, when most of the movies have lower sentiment score than audience rating score. Additionally, in the BERT model, one movie, which is Pirates of the Caribbean 4, has been predicted to match the predicted audience scaled rating. In the BERT model, most movies belong to the area of the diagonal line and the x-axis, which implies that their audience scaled ratings are higher than the predicted sentiment score. However, in the VADER model, most movies from Twilight sequels have the sentiment score higher than the audience score, while pirates sequel’s movies tend to have higher score in audience ratings. Only Pirates 4 in the VADER model in Pirates of the Caribbean’s sequel receives a higher score in sentiment prediction than its audience rating.

**Discussion**

Our results gave us some interesting insights, but also showed the challenges of doing sentiment analysis on Reddit comments. One clear takeaway is that VADER tends to provide more positive sentiment scores than BERT. This might be because VADER doesn’t fully understand context or sarcasm, which are common on Reddit, while BERT is better at catching these nuances. When comparing Pirates of the Caribbean and Twilight, we saw that Pirates generally had higher sentiment scores, especially in the earlier movies. However, even with higher budgets in later films, sentiment often dropped, which shows that more money doesn’t always mean better audience reactions. Another critical point is that while sentiment and box office success were somewhat related, they didn’t always match up. This means other things like timing, marketing, or fan loyalty probably play a significant role too. We also ran into several limitations. Some Reddit comments talked about actors, specific scenes, plotlines, or even general opinions about the franchise, but not about the movie as a whole, which made labeling tricky. Hence, building accurate sentiment models was difficult. We worked extensively to improve our results, trying different models and techniques, but the accuracy stayed lower than expected. Even though we labeled 300 comments per movie, sentiment is still subjective, and opinions often overlapped or contradicted each other. Plus, Reddit’s user base might not reflect the broader movie-going public. In the end, even though we improved our model using BERT and ensemble methods, accuracy was still not perfect. This shows how tough it is to accurately classify real-world, messy data. In the future, using more data and combining sentiment with metadata like release dates or trailers might help make predictions more reliable.

In summary, while our models improved and offered meaningful insights, this project highlighted how complicated and nuanced sentiment analysis becomes with real-world data. With better filtering, advanced modeling techniques, and more diverse data, future research could build on our work to create more accurate and context-aware sentiment analysis tools.

**Conclusion & Future Work**

The VADER model served as our baseline and consistently produced higher sentiment scores than our BERT model, indicating a tendency toward more positive interpretations. It showed slightly higher sentiment for the *Twilight* series compared to *Pirates of the Caribbean*, though this may reflect its sensitivity to enthusiastic language rather than true sentiment. While useful for quick analysis, VADER struggled with sarcasm, context, and nuanced opinions, as mentioned before, which is common in Reddit comments, limiting its effectiveness for complex sentiment tasks. Overall, the BERT‐based sentiment consistently falls below VADER’s estimates, indicating VADER’s positive bias. Across both models, the Pirates series maintained higher mean sentiment than Twilight, reflecting stronger fan engagement. Mean sentiment showed only a modest relationship with box-office and budget, suggesting story quality outweighs production spend in driving fan reactions. Fan‐derived sentiment scores tend to be slightly more reserved than critic and audience ratings. Additionally, it is interesting to note that early sequels (“Curse of the Black Pearl” and “New Moon”) captured the highest positivity, with later entries showing gradual declines. Our idea of combining transformer and classical models improved robustness, but highlights the inherent challenge of accurately classifying nuanced Reddit comments.

As future work, we could apply the pipeline to other long-running series to test whether these patterns hold. Additionally, we could integrate larger transformer variants such as RoBERTa or DeBERTa, or ensemble multiple contextual models for improved nuance. But also, augment data with release-timeline events (trailers, casting news) and basic user metadata to explain sudden sentiment shifts. RoBERTa, for example, is an optimized version of BERT that is trained with more data and better fine-tuning strategies, resulting in stronger performance on many NLP tasks. DeBERTa goes a step further by improving how word positions and meanings are encoded, helping the model better understand the relationships between words—especially useful for complex or informal text like Reddit comments. These more advanced models could help capture subtle sentiment variations more effectively than standard BERT.

Another direction would be to implement multi-label sentiment classification, allowing one comment to be tagged with multiple sentiments (e.g., both positive and negative), which would better reflect the layered and often mixed nature of real online opinions. We also recommend improving comment filtering by tagging them based on specific themes—such as character discussions, plot critiques, or production-related feedback—to allow for more focused analysis. Finally, expanding our dataset beyond Reddit to include comments from YouTube, Twitter, or fan forums could reduce bias and give a more comprehensive picture of audience sentiment.

By addressing these areas, future research could build on our foundation and improve both the accuracy and interpretability of sentiment analysis applied to media franchises.

**Acknowledgements**

We would like to thank the following for their support and contributions to this project:

* Professor Cheng-Yuan Ho, for guidance on study design, model selection, and manuscript preparation
* The TAs of this class who have shown support, and answered our multiple questions, guiding us in the right direction
* The Reddit community, for making their comments publicly available and thereby enabling this analysis
* Open-source developers of PRAW, pandas, scikit-learn, PyTorch, and Hugging Face Transformers, whose tools underpinned our entire pipeline.

**Statement of Contribution**

This project was a collaborative effort, and each team member played a key role in ensuring its success. The responsibilities were divided as follows:

**Data Collection**

For this phase, Maelys led the data collection, including sourcing our data from Reddit’s API, cleaning, and compiling relevant datasets used in the analysis. This was after we all discussed which social media platform we wanted to use.

**Data Preprocessing**

For this phase, Sora led the data preprocessing, handling tasks such as data normalization and feature engineering. Ewald worked on coding for our baseline model on Vader. All of us contributed to manually labeling our training data. Lillian worked on code that fixed misspellings and nulls in our labeled data so that all 10 datasets were uniform and ready for analysis.

**Model Development**

Sora took charge of model development, including selecting appropriate algorithms, training the models, and performing parameter tuning. We were all involved in the process of running the code, fixing errors, and coding to explore different models to see which one performed the best. Ewald coded for our baseline model, using VADER to label all of our 10 datasets. Lillian coded and combined all the individual VADER labeled data into one large dataset.

**Result Analysis**

For creating visualizations, Sora coded visualizations for all the movie sentiments. Lillian helped run the code and compile the results. Maelys coded visualizations for individual movie sentiments and Anh worked on running the code and compiling results. Anh also created visualizations for IMDB and Rotten Tomatoes ratings for both movie series. Sora collected the box office and budget data for both movie series. Ewald created visualizations for VADER sentiment predictions.

**Report Writing**

Maelys coordinated report writing, integrating all sections into a cohesive outline and ensuring consistency in the final write-up. However, we all contributed to writing down our processes and editing the outline to ensure accuracy and clarity. Maelys worked on the introduction, Lillian worked on related works, Ewald worked on discussion, Anh worked on data description.

**Presentation Preparation**

We were all responsible for presentation preparation, creating the slides, rehearsing delivery, and presenting the findings to the class/stakeholders.

All members contributed to project ideation, discussion, and decision-making throughout the process. We acknowledge each individual's contributions and are grateful for the collective effort that led to the successful completion of this project.

**References**

Batra, H., Punn, N. S., Sonbhadra, S. K., Agarwal, S., Kotsis, G., Strauss, C., Khalil, I., & Tjoa, A. M. (2021). BERT-based sentiment analysis: A software engineering perspective. In *Database and Expert Systems Applications* (Vol. 12923, pp. 138–148). Springer International Publishing AG.<https://doi.org/10.1007/978-3-030-86472-9_13>

Chang, M. W., Devin, J., Lee, K., & Toutanova, K. (2018, October 11). *[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv. Retrieved May 27, 2025, from https://arxiv.org/abs/1810.04805

Dhaoui, C., Webster, C. M., & Tan, L. P. (2017). Social media sentiment analysis: Lexicon versus machine learning. *The Journal of Consumer Marketing, 34*(6), 480–488.<https://doi.org/10.1108/JCM-03-2017-2141>

Haanaes, K. B. S. (2025, January 7). Déjà vu: Is the film industry’s sequel and remake addiction a sign of the end? *IMD Business School for Management and Leadership Courses*.<https://www.imd.org/research-knowledge/economics/articles/deja-vu-is-the-film-industrys-sequel-and-remake-addiction-a-sign-of-the-end/>

Hernández-Pérez, R., Lara-Martínez, P., Obregón-Quintana, B., Liebovitch, L. S., & Guzmán-Vargas, L. (2024). Correlations and fractality in sentence-level sentiment analysis based on VADER for literary texts. *Information (Basel), 15*(11), 698.<https://doi.org/10.3390/info15110698>

Hossen, Md. S., & Dev, N. R. (2021). An improved lexicon-based model for efficient sentiment analysis on movie review data. *Wireless Personal Communications, 120*(1), 535–544.<https://doi.org/10.1007/s11277-021-08474-4>

Manuel-Ilie, D., Gabriel, P. A., & George, C. R. (2023). Sentiment analysis using BERT model. *International Journal of Advanced Statistics and IT&C for Economics and Life Sciences, 13*(1), 59–66.<https://doi.org/10.2478/ijasitels-2023-0007>

Nkhata, G. (2022). *Movie reviews sentiment analysis using BERT* [Master’s thesis, University of Arkansas]. ScholarWorks@UARK.<https://scholarworks.uark.edu/etd/4768>

Pano, T., & Kashef, R. (2020). A complete VADER-based sentiment analysis of Bitcoin (BTC) tweets during the era of COVID-19. *Big Data and Cognitive Computing, 4*(4), 33.<https://doi.org/10.3390/bdcc4040033>

Samuel, S. T., & Marikannan, B. P. (2020). Sentiment analysis on movie reviews using Twitter. *Journal of Computational and Theoretical Nanoscience, 17*(7), 2869–2875.<https://doi.org/10.1166/jctn.2020.9326>

Teja, J. S., Sai, G. K., Kumar, M. D., & Manikandan, R. (2018). *Sentiment analysis of movie reviews using machine learning algorithms: A survey*. International Journal of Pure and Applied Mathematics, 118(20), 3277–3284.<http://acadpubl.eu/hub/2018-118-21/articles/21d/44.pdf>

White, A. (2024, September 23). Hollywood’s dramatic decline results in sequels – is this the end? *The Torch*.<https://www.valpotorch.com/arts_and_entertainment/article_b0d68e30-7701-11ef-91fe-e76fb9691cf0.html>

Youvan, D. (2024, June). *Understanding sentiment analysis with VADER: A comprehensive overview and application*. ResearchGate.<https://www.researchgate.net/publication/381650914_Understanding_Sentiment_Analysis_with_VADER_A_Comprehensive_Overview_and_Application>