Predicting the Rating of a Film or Television Show: A Study of Sentiment Analysis on YouTube Comments

Andreas Belsager, Mads Høgenhaug, Marcus Friis & Mia Pugholm
IT-University of Copenhagen
Data in the Wild (KSDWWVD1KU)
Wednesday 21st December, 2022

***Abstract*—This study investigates the relationship between the initial audience reaction to a trailer on YouTube and the subsequent success of a movie or TV show, as measured by the listed user rating on IMDb. To analyze this, we created a dataset using the YouTube API to retrieve videos and comments for trailers and the Return YouTube Dislikes extension to access likes, dislikes, and view counts. We also collected data from IMDb and applied sentiment analysis using a pre-trained Vader model to label all the comments, with a portion manually annotated as gold labels. The final dataset consisted of 757,768 comments from 440 trailers. We found a significant positive correlation between the initial reaction to a show or film's announcement and its listed IMDb rating, suggesting that the reception of a show or film's announcement can have an impact on its critical success. However, the imprecision of the sentiment classifier limits the accuracy of the results. Further research is needed to identify the specific factors contributing to this relationship.**

Data and code are available on GitHub
https://github.com/Marcus-Friis/data-in-the-wild

I. Introduction

When promoting a new television show or movie, an official trailer is often released to generate interest and build anticipation leading up to the release. The trailer is designed to tease audiences on the upcoming movie or show, and to create an initial reaction from audiences. This initial reaction can potentially have a significant impact on the success of the movie or TV show, both in terms of critical reception and viewership. In this project, we aim to investigate the relationship between the initial audience reaction to a trailer on YouTube and the following user rating of the movie or TV show on IMDb.

To investigate this relation, we create a dataset from scratch, specifically designed to analyse relations between trailers and the final product. For TV shows, the final dataset will only contain trailers for the first season, and not consecutive seasons. We do this based on the assumption, that later seasons automatically will receive a better initial reception, since the TV show has been renewed. To this end, we utilize YouTube's official API (Google, n.d.-c) to retrieve videos and associated comments for movie and TV show trailers. Since YouTube has deprecated the dislike functionality, we also collect data from the 3rd party YouTube extension Return YouTube Dislikes (Dislike, n.d.), which allows us to easily access likes, dislikes and view count. Additionally, we collect data from the extensive movie database IMDb (IMDb, n.d.), which grants us information about movies and shows and ratings. In practise, the collection and merging of these data sources introduce a plethora of challenges, all of which this paper aims to solve. Due to practical limitations, we only collect data from a limited number of networks, namely Amazon Prime, Disney+, HBO and Netflix. This is due to quota constraints with the YouTube API, and doing it this way simplifies the querying of YouTube data.

To gauge opinions about the movies and TV shows at hand, we apply sentiment analysis to measure the mean sentiment for each trailer. Specifically, we use the pretrained sentiment classification model, Vader (Hutto & Gilbert, 2014), which is trained on a large Twitter dataset, to label all the comments. Additionally, we manually annotate a portion of the comment data, which yields gold labels to measure Vader's performance on the YouTube comments domain.

With the aforementioned techniques, we collect a dataset consisting of data from YouTube, Return YouTube Dislikes and IMDb, with labeled sentiments. The final dataset consists of $757,768$ comments spread across 440 trailers. This dataset is the basis of our analysis, delving in to what effect public opinion has on the performance of a movie or TV show.

A. Research question

How does the reception of a movie or TV show impact its user rating?

- Can sentiment analysis of YouTube comments accurately predict the user rating of a movie or TV show on IMDb?
- How does the sentiment of YouTube comments for a trailer differ between highly or lowly rated movies or TV shows on IMDb?

II. Data collection and processing

The focus of this study is on the process of collecting and processing data. To effectively reach an answer to the research question, it is necessary to know what data is needed, how to

obtain it, and how to process it. When it comes to analyzing the popularity of a TV show or movie, there are several pieces of data that are particularly useful. For this project we need data about the movies and shows themselves, as well as a large amount of opinions regarding these movies. For the former, we use IMDb, but the latter is a bit more challenging. Ideally, we would use data from an unbiased platform where people express opinions about movies and shows before they are released. However, there is no such platform. Instead, opinions could be collected from Twitter; a platform where users often express opinions with a large user base.

The use of Twitter for this project faces one central challenge; we only want opinions about movies and TV shows, and we need to map each opinion to its corresponding movie. This task is next to impossible to do on a large scale, since there is no standardized way of finding all opinions on one piece of media. Some have hashtags associated with them, which makes this task easier, but we would still need to map all hashtags to their corresponding movie or TV show. In short, while we would ideally use Twitter for this project, it is in practise impossible. Instead, we opt for an alternative, which is collecting comments from movie's or show's YouTube trailer.

Furthermore, by using YouTube trailers, we can measure popularity of the product by looking at metrics such as views, likes, and dislikes collected through a third party extension called *ReturnYouTubeDislikes*. We collect comments from each trailer and use them to gauge opinions about the displayed product.

A. YouTube API

In this study, we use the official YouTube API (Google, n.d.-c) to collect YouTube videos and their corresponding comments. This is a restful API, which means all interaction between client and server is stateless. This has some practical implications such as the need for pagination (RestfulApi, n.d.). We use Python's wrapper library 'google-api-python-client' to interact with this API (Google, n.d.-a).

One central challenge when working with this API is the limited units. The API has a default quota allocation of 10,000 units per day (Google, n.d.-b), which necessitates the collection of data over multiple days. To circumvent this limitation, we used multiple keys when collecting data, yet even with 4 keys, it still took approximately 4 days to collect the necessary data. The data collection process starts with collecting trailers.

1) Trailers: The trailers needed for this data are official trailers from the original creator and/or distributor released before the release of the advertised product. This constraint enforces that we need trailers to be the true official trailer, and not a knockoff, a re-upload or any other variation. To accurately enforce this constraint, we strictly collect trailers from trusted network's official channels. This means we only collect data from Amazon Prime, Disney+, HBO and Netflix.

For collecting the trailer videos from YouTube, we use *search* method (Google, n.d.-c) to list all videos corresponding to some search criteria. This operation costs 100 units (1 API key grants $10,000$ units per day), which strongly limits how many videos we can collect. In this search call, we set the query term "q" to "trailer" and also enforce that the results have to be from one of the 4 networks. Without this parameter, we would get random unofficial trailers, which is why we limit ourselves to the 4 channels. We then loop through all the response pages from the API with pagination.

Once all the data is collected, it must be cleaned, since YouTube returns more results than needed. For instance, "That's Queen to you #shorts" and "The Cast Of The Sex Lives of College Girls Play Acceptable Or Dealbreaker" are both listed by the search method when searching for "trailer" from HBO's official channel. We filter the collected trailers by enforcing a set of rules for all video titles e.g. they must contain "official" and only be trailers for the first season. These rules are based on looking at the frequency of unigrams in the videos' title (see figure 8 in appendix).

2) Comments: Once the trailers are collected, we scrape all comments to each trailer. This is done using YouTube API's *commentThread* method, where we list all items. Each list call costs 1 unit. To collect all comments, we iteratively call this method on each video, utilizing pagination to get all comments per trailer. It is necessary to scrape all comments, since we are primarily interested in comments posted before the release of the advertised movie/TV show. The method supports ordering the list in two ways: by time from newest to oldest, and by relevance. We are interested in the opposite of time, and relevance has little value to us. As such, we scrape all comments for each video, to ensure we get the oldest comments.

B. Return YouTube Dislikes API

In November 2021, YouTube removed the dislike functionality. For this project, since we are measuring sentiment regarding movies and TV shows, the dislike count is a valuable metric. A workaround solution is using the 3rd party YouTube extension called *Return YouTube Dislikes*. It works by using archived dislike data and dislike estimates based on user behaviour from users with the plugin installed (Dislike, n.d.). With this extension, we can access the dislike data through its API and thus get likes, dislikes and views for each trailer. We get views and likes through this API as well for consistency, and acquiring this information through YouTube's official API would cost us API units.

C. IMDb Data

To research the relationship between reception and reaction, we need data about the release dates, titles, ratings etc. of these

TABLE I: IMDb data head, see full version in table III in appendix

	title.basics			title.ratings		scraped dates
tconst	*titleType*	*primaryTitle*	...	*averageRating*	*numVotes*	*releaseDate*
tt0112159	tvSeries	Neon Genesis Evangelion	...	8.5	70345	20 August 1997
tt0489974	tvSeries	Carnival Row	...	7.8	63719	30 August 2019
tt0499097	movie	Without Remorse	...	5.8	58538	30 April 2021
tt0800325	movie	The Dirt	...	7.0	50280	18 March 2019
tt0805647	movie	The Witches	...	5.3	41745	22 October 2020

movies and TV shows. One way of getting this information is through IMDb.

IMDb (Internet Movie Database) is a website that provides information about movies, TV shows, actors, and other aspects of the entertainment industry. It is one of the largest and most comprehensive databases of its kind, and it is a popular source of data for researchers in various fields.

IMDb is a valuable resource to this research, since we are interested in studying the relationship between reactions to media and their critical reception. The website allows users to rate and review movies, TV shows, and other productions, providing a wealth of data on the opinions and reactions of audiences. This data can be useful for studying how different productions are received by audiences, and how these reactions compare to the critical reception of these productions.

IMDb does not offer an API for research, but instead provides all its' data in a downloadable format (IMDb, n.d.). They provide a dataset that is updated daily with the latest IMDb data. It has almost all the information that is available on the website. This dataset is split up into multiple files. For this project, we use the ones titled *title.akas.tsv.gz* and *title.ratings.tsv.gz*, which contain general information regarding movies, shows and more, as well as ratings for each. The data does however lack the exact release date that is otherwise available on the website. It is crucial for this project that we have this date, since we are interested in comments written on the trailer before the actual release of the movie/show. We assume that comments written after the release date of a media more often reflect the opinion of a person who has seen it, whereas we know for certain that comments written before the release date reflect the writers reaction without the influence of the media being released. This information is available on the IMDb website, but it is not included in any of the downloadable files. Ideally, IMDb would offer this data in the data files or with an API, but since they do not, we need to find another solution.

D. IMDb exact release dates

As a work around solution to the missing dates, we scrape them from the website. Since we already have the id's of all movies and TV shows from the IMDb downloadable dataset, we access the */title/tconst/releaseinfo* subsite of IMDb, which contains information about release dates for a lot of countries. The scraping is implemented using *BeautifulSoup* to parse the DOM and extract *HTML* elements (Richardson, 2007).

There are generally 3 approaches to scraping this date:

1) Get the earliest date.
2) Get the release date in US.
3) Get the most frequent date.

Approach 1 ensures that no comments in the final dataset will be from people who have seen the movie/show. However, many movies/shows are screened at film festivals or similar events, leading to release dates listed on IMDb potentially months before the actual public release date.

Approach 2 ensures consistency in the release dates. However, in some instances, movies/shows are released to the public earlier in other countries or regions, and thus, the dataset could include comments from people who have seen the product.

Approach 3 gets the date that most people will be able to access the media from. However, this might not always be the earliest possible date, leading to some user comments potentially having seen the product. However, since this is the most frequent date, the amount of these comments should be minimal. For this reason, and by trying different methods empirically, we found this to be the best approach.

After downloading the IMDb data and scraping the release dates, the data is ready to be joined. When joined, the IMDb data looks as seen in table I

E. Merging IMDb and YouTube data

Once the IMDb- and YouTube trailer data has been collected and prepared, the two need to be joined. This task is non-trivial, since we primarily have the titles and also some metadata to match them up on. There are 2 central challenges in this task

1) Title extraction: From the YouTube trailer title, the true title of the movie or show needs to be extracted. There is no standard enforced format for trailer titles, so this needs to be programmed or extracted with methods inspired from named entity recognition. A trivial approach would be to remove all stop words with custom YouTube-trailer-specific stop words added such as "official", "trailer", "teaser", "Netflix" etc., and all text after the first pipe |. However, some of these stop words could occur within the title, and removing them would lead to removing words from the title.

2) Title collisions: It is not uncommon for IMDb entries to share title with other entries. For instance, there are 69

distinct movies or TV-shows titled ”Home” within the IMDb database. Counting entries in total, the number of ”Home” titles become 685. This means that if e.g. Netflix released a trailer called ”Home”, the mapper would need to find the one correct option of the 685 possibilities within the IMDb data.

The combination of these two challenges falls outside the scope of this project. It would be essential for doing this study at a larger scale, but with the limited amount of trailer data collected from YouTube, we instead manually mapped each YouTube trailer to their respective IMDb entry.

F. The full dataset

Once all the separate pieces of data have been collected, it is joined into the full dataframe. Additionally, using the YouTube dataset and the IMDb dataset, we create a variable called *commentDateOffset* which represents the number of days between the release date of a movie or TV show and the date of a comment on YouTube. A detailed description of the data can be found in appendix table IV. This full dataset is used for all the following analysis.

There are many aspects of this data; this section will explain and visualize the most important of them. In the full dataset, the unique index is the commentId. Each row will correspond to exactly 1 YouTube comment. The data contains 757,768 comments spread across 440 trailers. In figure 1, the distribution of trailers across networks can be seen.

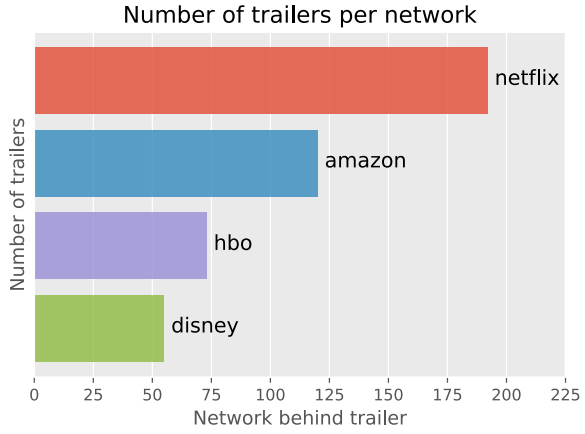

Fig. 1: Number of collected trailers per network in the final dataset. The number of trailers per network is upper bounded by the number of trailers posted on the network's YouTube channel, which is why the distribution is not equal.

These trailers have very different number of comments, and the distribution of comments per trailer somewhat follows a log-normal distribution (see figure 2). 70% of the trailers have less than 100 comments in total, making some analyses difficult or unreliable, since there are not that many comments.

Throughout this project, we make the fundamental assumption that people use YouTube comments on trailers to express their opinion about a certain product. However, this assumption may not be entirely true. An alternative metric to comment sentiment could be YouTube likes and dislikes. We gathered this data through the Return YouTube Dislikes API (Dislike, n.d.), and with it, we can investigate the relationship between comment sentiment and like-dislike ratio.

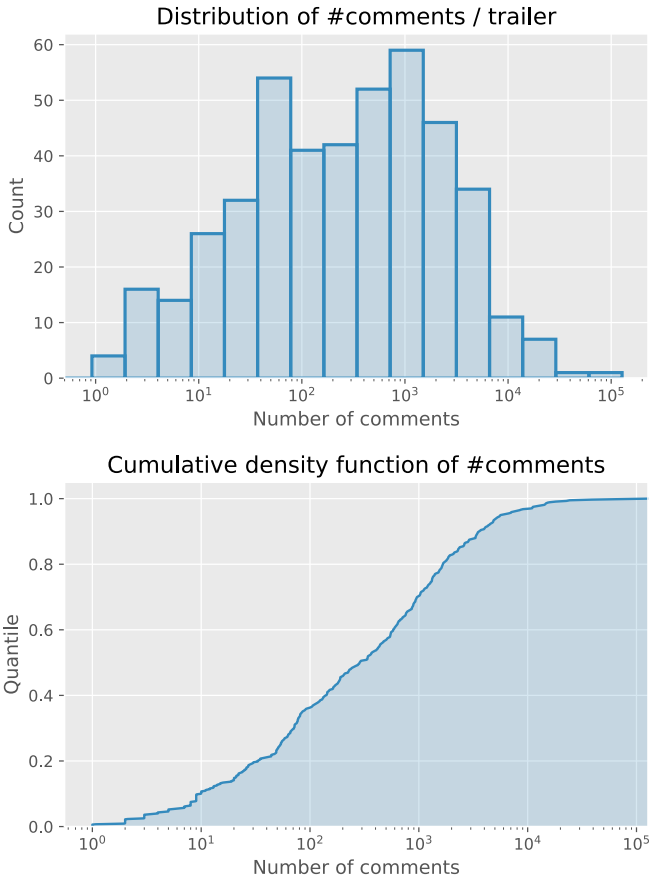

Fig. 2: Distribution of number of comments per trailer. On the histogram, we see that the distribution is somewhat log-normally distributed, meaning we have a long tail of trailers with a lot of comments. The cumulative density distribution verifies this, showing that approximately 70% of trailers have 10^3 comments or less.

On figure 3, we see that there is a slight positive correlation between the like-dislike ratio and the average sentiment per trailer, but nothing major. Concretely, calculating the Pearson Correlation Coefficient and doing a non-correlation test (Virtanen et al., 2020), we see that the correlation $r = 0.17$ with $p = 0.0004$ is slightly positive, yet statistically significant. If we assume like-dislike ratio accurately reflects users opinion about a trailer (which most likely is an incorrect assumption), it could indicate that average comment sentiment might not be the most ideal metric for gauging interest and opinion. However, we cannot access historical data for likes and dislikes in the same way we can with comments, which leads us to using comment sentiment for this project, since it is imperative

we can gauge opinion before the release of the movie/show.

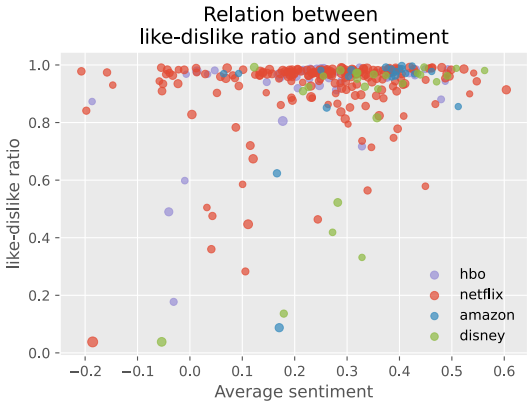

Fig. 3: Scatter plot visualizing the relationship between the average sentiment of comments per trailer and the like-dislike ratio. Note how even though the ratio is 1, the sentiment still varies greatly.

The target variable for the analysis, as framed by the research question, is the average user rating listed on IMDb. As seen in figure 4, this variable is approximately normally distributed. There is a weak correlation with the number of votes $r = 0.20, p = 2.52e - 05$, and no correlation between the rating and log number of user ratings. This could indicate that just because a movie or show garners a lot of attention, we should not expect a particular rating, yet there is some linear positive correlation. This information is vital, since, intuitively, we would expect the number of comments on a trailer to reflect, how many people are interested in the product, and how many people review it.

III. Methods

A. Data annotation

This section describes the process by which we labeled and classified the YouTube comments that we collected for our study. This involved creating detailed annotation guidelines as well as manually reading and evaluating a subset of comments to determine their sentiment, and then assigning each comment a label of positive, negative, neutral or not applicable.

We started by creating an annotation guideline (see section VII-A), which details all the instructions necessary for annotators to start labeling. In it, we describe how the default label should be *neutral*, and when to use the *positive* and *negative* labels. Since our data is unclean, we also created the label *not applicable*, which should be used whenever we "don't want the comment". Concretely, it should be used whenever a comment is in a foreign language, contains spam, is an advertisement, contains a link etc.

For the actual annotation process, we use Label Studio (Label-Studio, n.d.). It provides a framework for labeling text in a web-hosted user interface. It allows creators to define a

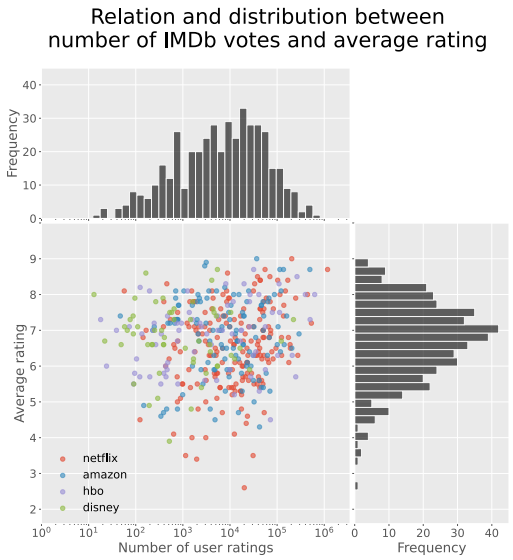

Fig. 4: Scatter plot with marginal distributions of number of user ratings and average rating. The scatter shows a slight positive correlation, which could be interpreted as the popularity of a movie/show influencing the average rating.

custom *HTML* site, which the annotator interacts with. For this paper, we used Label Studio's sentiment classification template, to which we added the *not applicable* label. By using Label Studio, we streamline the annotation process, while minimizing human error.

We used 3 annotators in total. When the annotation session began, the annotators started by going over a few examples together to align their preconceived notions of sentiment. The annotators in total annotated 1688 times across 1417 unique comments. Additionally, 120 were randomly sampled, which all the annotators were required to annotate (see figure 9 in appendix for the agreement visualized). This lets us investigate the agreement between annotators using Cohen's Kappa (McHugh, n.d.) and estimate the difficulty of the task and the quality of the guidelines.

To evaluate the level of agreement between the different annotators, we used Cohen's Kappa to calculate the inter-annotator agreement. The Kappa value that we obtained was 0.75, indicating a substantial level of agreement among the annotators (McHugh, n.d.). This suggests that the annotations made by the annotators are consistent and reliable, and that the labeling process has produced results that are likely to be accurate and reflective of the underlying data. Furthermore, the Kappa value of 0.75 also indicates that the agreement between the annotators is not simply a result of chance, but rather reflects a real and meaningful level of consensus among them. The Kappa value is highly dependent on the fact that the three annotators annotated 95/120 comments with the

same label, which tells us that many of the comments carry a general consensus which are easy to identify. However, some of the comments proved to be challenging to annotate, leading to instances of disagreement.

B. Sentiment classification model

To classify the sentiment of the YouTube comments, we use the `vaderSentiment` library (Hutto & Gilbert, 2014), a rule-based sentiment analysis tool. Vader (Valence Aware Dictionary and sEntiment Reasoner), in particular, is specifically designed to analyze sentiment in social media text data, which tends to be more informal and abbreviated than other forms of text. However, one potential drawback of using this model is that it may not always accurately identify sarcasm or irony in text. This is because vaderSentiment uses a lexical approach, which can struggle to identify the negation or reversal of words that are often used in sarcastic or ironic statements. This is a general challenge in NLP, and since this paper is not focused on state-of-the-art NLP methods and tools, we use it anyway. It is also important to recognize that the model was trained on Twitter data, which may differ somewhat from the language used in YouTube comments. As such, it is reasonable to assume that the model's performance on tweet-like texts may not fully generalize to YouTube comments. Ideally, we'd either use another sentiment classification model or, even better, fine-tune a model using our already annotated data to perform better on this particular domain. Nevertheless, fine-tuning is outside the scope of this project.

The sentiment classification model is evaluated on our domain using precision, recall, and F1-score (Géron, 2019, p. 92) for the three labels: positive, neutral, and negative. Table II shows that the model performed the best on the positive label, followed by the neutral and negative labels. A high precision for positive and neutral comments means that the model is able to identify a large proportion of positive and neutral comments correctly. However, the model demonstrated low recall for the neutral label, suggesting that it tends to predict other sentiments when the true sentiment is neutral. The F1-score, which is a balance between precision and recall, was highest for the positive label and lowest for the negative label. The counts shown in the table indicate the total number of predicted labels in each category, and we can see that there is a low number of negative comments compared to positive and neutral.

This is a very poor performance, and it has huge ramifications for the result of the upcoming analysis. It introduces a level of imprecision that could potentially skew the entire analysis, and thus, the results should be taken with a grain of salt.

TABLE II: Performance of Vader sentiment classification model on YouTube comments domain, evaluated with annotated data.

	Precision	***Recall***	***Fscore***	***Count***
Positive	0.62	0.69	0.66	563
Neutral	0.60	0.47	0.53	514
Negative	0.40	0.52	0.45	187

C. Data analysis

For the analysis, we apply 3 approaches to find an answer to the research question.

1) Time series analysis: To figure out the relationship between average sentiment, rating and time, we frame the problem as a time series, where each trailer has a mean sentiment score for every day. With this approach, we can investigate how the sentiment evolves over time between two (or more) groups. We create these groups by splitting the dataset based on a threshold. We use the median of IMDb score as the threshold, and offset the two groups by 0.6, increasing the gap between the two groups. We also remove all trailers with less than 100 comments, to ensure the sentiment reflects the opinion of a representative sample of people. We are left with two groups with approximately 120 trailers in each group. This ensures that the two groups are drastically different with regards to their IMDb score, and if there is a difference between these two groups, the visualization should reveal it.

As the time dimension, we use the number of days before/after release, e.g. if a movie is released 19-12-2022 and a comment on the trailer is posted 16-12-2022, the time dimension would be -3, since it is 3 days before release. This enables filtering of comments posted before and after release of a movie/show and facilitates easy comparison between all comments written n days before or after release.

The y-dimension is calculated as the median of the mean trailer sentiment for all trailers at a given timestamp. We use the mean sentiment score per trailer as a metric for the general sentiment of users. This is applied on a daily basis, and the median of all these values are taken to represent the general case.

To detect trends in the data, we apply a rolling window technique with a window size of 7 days to smooth the series. For visual purposes, this window is not centered, since it would visually make it seem like a trend occurs before release, when in fact, it occurs after release. This method allows us to detect any long-term trends in the data, such as an increase or decrease in sentiment over time for trailers with high and low ratings on IMDB. With this method we smooth any fluctuations in the data that may be caused by a low number of comments and instead focus on the underlying trends.

The process described above yields a visualization showing the evolution of sentiment on trailers over time for highly and lowly rated movies and shows. The figure can be seen in figure 6.

2) Scatterplot: The relation between sentiment on trailers and IMDb rating can alternatively be explored as a scatterplot. Two scatter plots are created, one for sentiment before release date and one after. Plotting them side by side, we visualize the change in sentiment that happens once the corresponding movie/show is released. We add a best linear fit to both plots, with the intention of visualizing the general trend. Beside the linear fit, we display the Pearson correlation coefficient, which indicates how valid the trend is. The Pearson correlation coefficient was calculated using the *scipy.stats.pearsonr* library (Virtanen et al., 2020)). The resulting p-value was used to determine the statistical significance of the correlation. A p-value less than 0.05 indicates that the correlation is statistically significant.

3) T-test: For investigative purposes, the distributions of ratings for trailers with high and low average sentiment, we conducted a t-test. For this test we only used the trailers with more than 100 comments. The two groups were created by splitting the dataset based on the median of the average sentiment score for all trailers. Once the groups were created, we use t-test to statistically compare the distribution of the two groups and determine if the difference is statistically significant. Concretely, the t-test tests for the null hypothesis that two samples have identical expected values. For this application, a rejection of the null hypothesis would mean the mean IMDb ratings are statistically different based on the initial reaction on the trailer from the comments aggregated sentiment.

IV. Results

A. Time series analysis

The time series analysis showed that the low rated movies/TV shows had a decrease in sentiment over time, while those with high ratings had a more stable development (see Figure 6). After the release of the product, notice how the two groups diverge in a manner that reflects the order of their ratings.

Furthermore, we see that the development of median number of comments is decreasing after the upload of the trailer, but increases shortly before the movie/TV show is released. After the release, movies/TV shows with high ratings consistently receive more comments than movies/TV shows with low rating.

B. Scatterplot

Our analysis of the relationship between the average sentiment of comments on a trailer and user rating on IMDb revealed a correlation of 0.13 for comments made before the release of the movie or TV show, and 0.25 for comments made after the release (see Figure 7). This suggests that there is a slight, positive relationship between the two variables. The corresponding p-values were found to be 0.0409 and 0.0002, respectively, indicating that the correlation is statistically significant at the 0.05 level. Overall, our results suggest that there is a weak, but statistically significant, positive relationship between the average sentiment of comments on a trailer and user rating on IMDb, both before and after the release of the movie or TV show.

C. T-test

With an obtained p-value $p = 0.028$, the t-test showed that the difference in mean IMDb ratings, between the two groups, was statistically significant, given our p-value is less than 0.05. This means that trailers with higher average sentiment scores have a statistically higher expected IMDb rating than the ones with poor sentiment score as seen in figure 5.

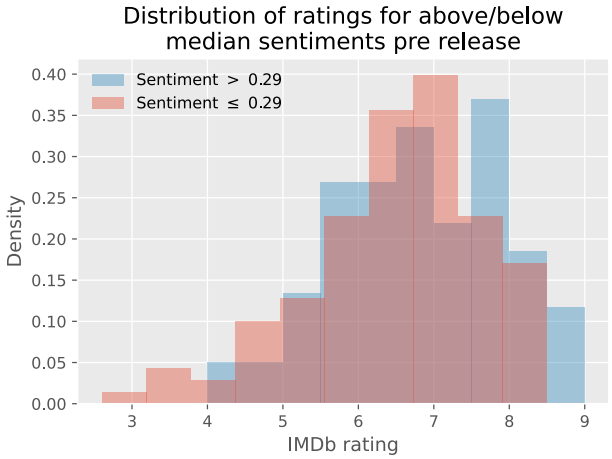

Fig. 5: Visualization of the groups used for the t-test, showing the distribution of IMDb ratings for above and below average sentiments.

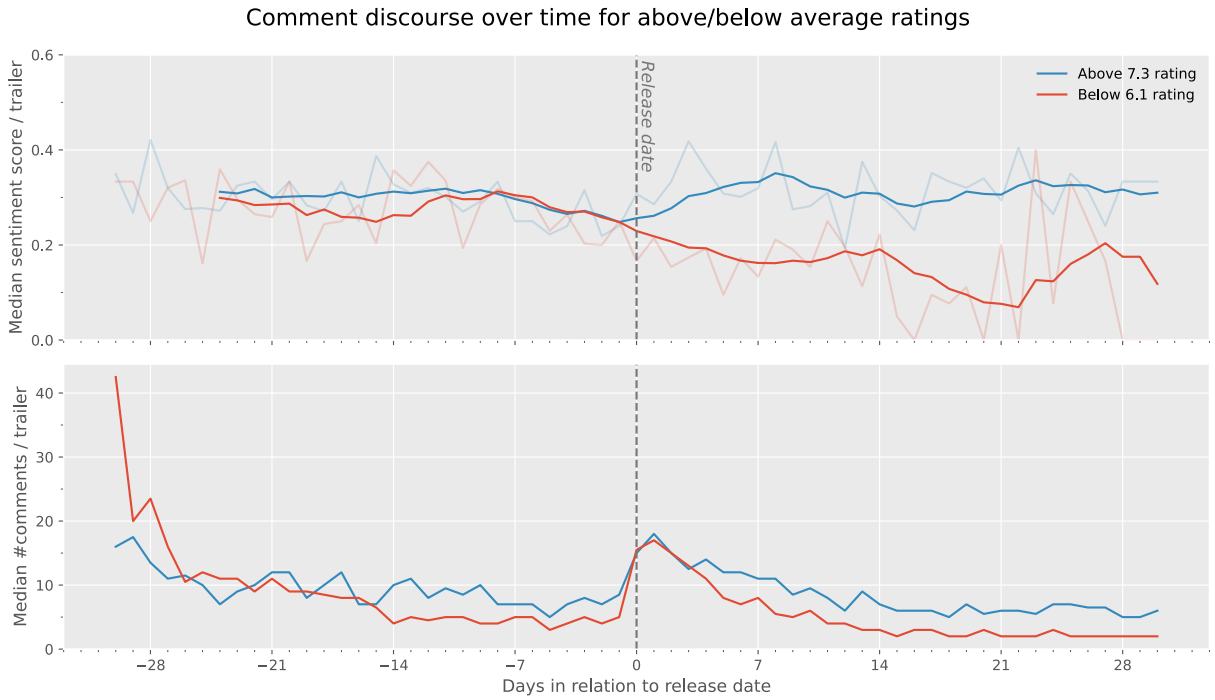

Fig. 6: Aggregate timeseries for the development of sentiment of highly and lowly rated movies/TV shows. It shows how, pre release, the sentiment between the two groups is largely the same, yet when the product is released, the trailers gain a boost of activity again and the sentiment diverges to reflect the order of the ratings.

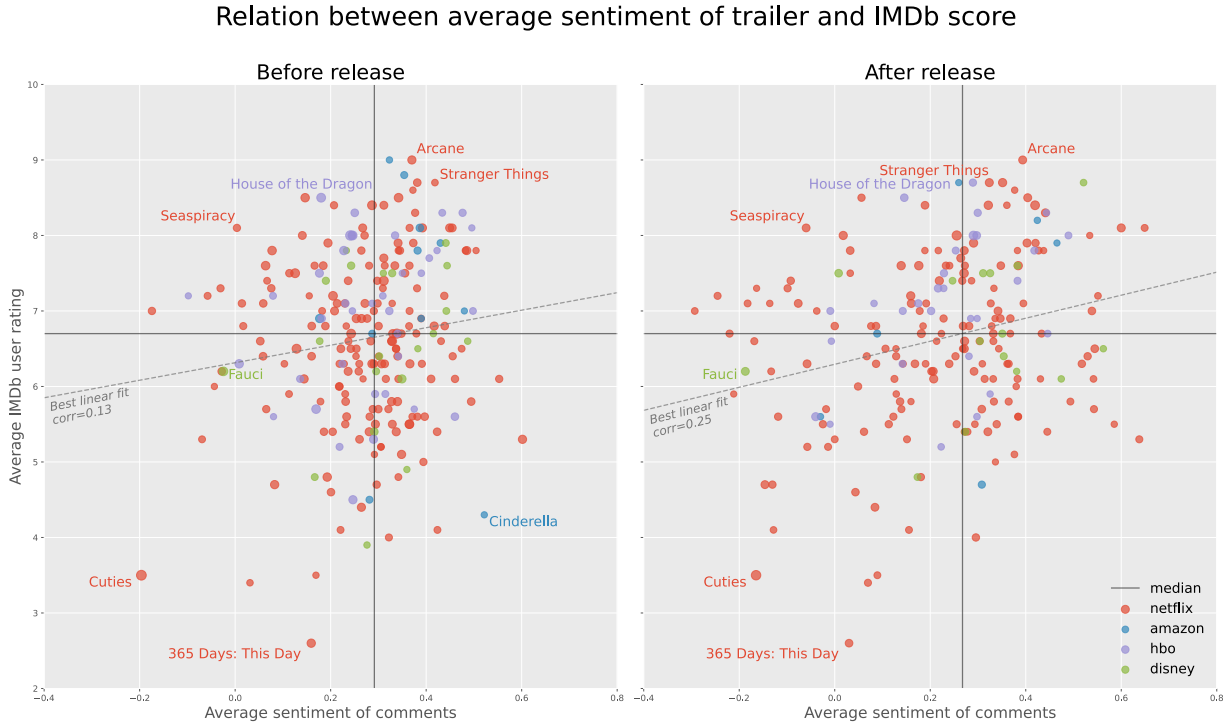

Fig. 7: Comparison of the relation between sentiment and IMDb rating for each trailer. Notice how, pre release, there is a slight positive correlation, but after the movies come out, this correlation increases. This indicates that the sentiment before release has a weak relation to the IMDb rating, but this relation increaes when measuring comments sentiment after release.

V. Discussion

A. Sentiment classification

The biggest issue with this study lies within the inaccuracy of the used sentiment classifier. The analysis can only be as accurate as the data it builds on, and without reliable sentiment labels, the whole analysis becomes unreliable.

One potential reason for the relatively low precision and recall for negative and neutral comments could be the presence of sarcastic, ironic or emotional statements in the test dataset. As mentioned in section III, vaderSentiment struggles to identify and correctly classify the sentiment of these types of statements due to its lexical approach. Additionally, the classifier was trained on Twitter texts and not YouTube comments, which most likely also impacts the accuracy negatively due to the domain shift. Therefore, the library has assigned incorrect sentiments to a significant portion of the comments in the test dataset, resulting in a lower overall performance. For instance, a comment like *"I'm not crying, you are *crying emoji*"* may be classified as negative due to the presence of the word "crying," despite the overall sentiment being positive given the context of the trailer. In future research, it would be beneficial to train our own classifier based on a dataset of YouTube comments to improve the performance of the sentiment analysis. This would allow the model to be more tailored to the specific language and tone used in YouTube comments, resulting in a higher performance.

One large drawback of using sentiment as the classification task is, that it may not accurately capture the overall sentiment of the comment with regards to the displayed movie/TV show, leading to potential inaccuracies in our data. To address this challenge, we may want to consider using excitement as the classification task instead of sentiment. By using excitement, we can better capture the enthusiasm of the person who wrote the comment, regardless of whether they used positive or negative language to express it. This could assist us in more accurately categorizing the comments and enhance the quality of our labeled data. However, we will need to carefully consider the specific guidelines and instructions for annotating excitement in our data. An example of where this could be beneficial is for Netflix's popular documentary "Seaspiracy". This movie is highly rated, yet the sentiment of the comments are very negative. This is largely due, not to lack of fondness of the movie, but due to the negative subject matter. Comments such as *"I was born in Fiji, this means I grew around the ocean. I cannot explain what I feel for the ocean but it's something of very valuable to me. I can't imagine my life without it. Whenever I am having a bad day -the ocean can fix it. Seeing it exploitated hurts me, it really does"* are negative in sentiment, but the person liked that the movie brought it to people's attention.

B. Data collection

The use of YouTube's API to retrieve data provided a convenient and efficient method for gathering a large volume of data on initial reactions to movie and TV show trailers. However, it is important to consider the limitations of this data source.

One potential issue is that YouTube may not be the most reliable source for capturing the true feelings of the audience. As seen in the results we found a majority of positive reactions in the comment section. However, it is possible that there may be a bias towards more positive comments on YouTube, as people who are excited about a show or film may be more likely to leave a comment than those who are indifferent or dissatisfied. For this reason, the overall sentiment on the platform may not accurately reflect the true feelings of the audience.

There is also the issue that the vast majority of trailers do not gain a lot of attention, particularly after release. Looking at the timeseries figure 6, we see that trailers generally gain a lot of attention when the trailers come out, and when the movie/TV show comes out. However, other than these times, the average trailer do not receive a lot of attention, which results in low sample sizes when investigating the sentiment in relation to time.

Furthermore, relying solely on YouTube data may not provide a complete picture of audience reactions. Other social media platforms such as Facebook or Twitter, could potentially offer a more diverse and representative sample of the audience. Therefore, it would be beneficial to complement the data from YouTube with data from other sources in future studies.

C. Upscaling the study

To improve this study and its results, it could be beneficial to acquire more data. However, this is heavily bottle-necked by YouTube's API quotas. If these could be ignored, a next iteration of this study would scrape more trailers from other major networks such as Hulu, Warner Bros. Pictures, Sony etc. This could yield a large increase in the number of collected trailers, and thus, the analysis would be based on larger samples. Larger sample sized would make the analysis more conclusive. Additionally, the used networks are all popular streaming services, and since we do not compare to any non-streaming service trailers, we do not know if there is an inherent bias within these types of trailers.

D. Annotation

One challenge we faced was the difficulty in accurately labeling quotes. A potential issue with labeling quotes is that they can have a negative sentiment, but be written with a positive intent by the author of the comment. For example, a quote from a movie or TV show that expresses a negative sentiment may be included in a comment by a fan of that movie or TV show. In this case, the quote itself may have a negative sentiment, but the author of the comment may have a positive sentiment towards the movie or TV show, and therefore, their overall comment may have a positive sentiment. This can make it difficult for annotators

to accurately determine the sentiment of a quote, as they must take into account the context in which it is used, as well as the intent of the person who wrote the comment. A second challenge is hashtags. Like quotes, hashtags can also be difficult to label correctly because they can be used in different ways and have varying levels of sentiment. For example, a hashtag can be used to express the sentiment of the person who wrote the comment, or it can be used as a form of irony or sarcasm.

In order to address these challenges, we plan to include specific guidelines in our annotation guide for how to handle quotes and hashtags for a second round of annotation. However, a second round of annotation was beyond the scope of this project. For future research, by providing clear and detailed instructions for annotating quotes, we hope to see an improvement in the consistency and accuracy of our labeled data which would lead to sentiment analysis algorithms are being trained on high-quality training data.

Another problem we might not be able to clarify in the guidelines, are how to handle sentences with mixed sentiment. E.g., the comment *"these are literally the worst movies but they're so addicting for what"* presents a challenge for annotators because it contains both positive and negative sentiment. On one hand, the phrase "these are literally the worst movies" suggests a negative sentiment towards the movies in question. On the other hand, the phrase *"they're so addicting for what"* suggests a positive sentiment, as the person who wrote the sentence finds the movies to be addictive despite their poor quality. This type of mixed sentiment can make it difficult for annotators to accurately determine the overall sentiment of the sentence.

VI. Conclusion

This paper set out to study the relationship between the sentiment of comments written on trailers and the rating of the corresponding movies. To this end, we collected data from YouTube and Return YouTube Dislikes through their respective APIs, from IMDb's premade datasets, scraped the IMDb website for release dates, and merged all the data together to create a dataset, which allows for investigating the relation between YouTube trailers and IMDb ratings.

The result showed that there is a statistically better average sentiment within the comment section of trailers before a movie or TV show is released for movies with higher ratings than more poorly rated movies. After the release of a movie or TV show, the results show that the sentiment of the comment section generally better reflect the IMDb rating of that particular media. This is evident when treating the comments as timeseries and when investigating the relationship between average comment sentiment and IMDb rating before and after the media is released.

These findings are heavily limited by a lackluster sentiment classifier performance and a relatively small amount of data. To improve these findings, the sentiment labels would have to be improved, either by fine-tuning a sentiment classifier on the YouTube comments domain, or changing classification task from sentiment classification to engagement classification. On top of that, the scale of the study would ideally be increased, branching out to include other networks and potentially even new data sources such as Twitter.

References

Dislike, R. Y. (n.d.). Return youtube dislike [Accessed: 15/12-2022]. https://www.returnyoutubedislike.com/

Géron, A. (2019). *Hands-on machine learning with scikit-learn, keras tensorflow*. O'Reilly.

Google. (n.d.-a). Google api client [Accessed: 12/09-2022]. https://github.com/googleapis/google-api-python-client

Google. (n.d.-b). Youtube api quota usage [Accessed: 12/09-2022]. https://developers.google.com/youtube/v3/getting-started#quota

Google. (n.d.-c). Youtube api v3 [Accessed: 12/09-2022]. https://developers.google.com/youtube/v3

Hutto, C., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the international AAAI conference on web and social media*, *8*(1), 216–225.

IMDb. (n.d.). Imdb datasets [Accessed: 10/12-2022]. https://www.imdb.com/interfaces/

Label-Studio. (n.d.). Label studio [Accessed: 12/12-2022]. https://labelstud.io/

McHugh, M. L. (n.d.). National library of medicine [Accessed: 14/12-2022]. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3900052/

RestfulApi. (n.d.). Restfulapi [Accessed: 20/12-2022]. https://restfulapi.net/

Richardson, L. (2007). Beautifulsoup documentation. *April*.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2

VII. Appendix

Find all code and resources here:
https://github.com/Marcus-Friis/data-in-the-wild

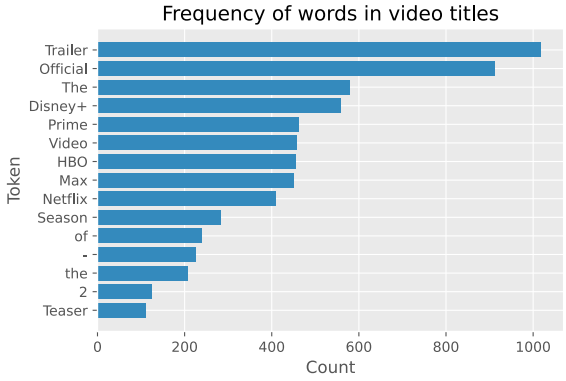

Fig. 8: Count of top 15 unigrams in YouTube video titles for results queried with ”Trailer”.

A. Annotation guidelines

This guide introduces you to the task of sentiment classification. It will give you the instructions you need in order to make sure you do everything correctly.

You are given a set number of comments. Your task is to determine whether these sentences carry positive, negative, or no sentiment.

Oxford dictionary defines sentiment as: *“a feeling or an opinion, especially one based on emotions”*

Concretely, you should use the following labels when:

- **Positive sentiment:** Comments that expresses a positive emotion or opinion. Examples include tweets that say ”This trailer is epic!” or ”This movie was amazing!”

- **Neutral sentiment:** Comments that are objective or neutral in tone. It could for instance be a comment that provide factual information without expressing a clear opinion, e.g. ”The weather today is sunny and warm.”

- **Negative sentiment:** Comments that expresses a negative emotion or opinion. Examples include ”This trailer is not particularly good in my humble opinion” or ”I hated that movie.”

- **Not applicable:** Comments that contain foreign languages, links or spam, e.g. malicious website links, and advertisements.

In cases of ambiguity or uncertainty, annotators should label the text as neutral. This includes cases where the sentiment is mixed or the text is difficult to interpret.

TABLE III: IMDb data head with all columnms

	title.akas								title.ratings		scraped dates
tconst	*titleType*	*primaryTitle*	*originalTitle*	*isAdult*	*startYear*	*endYear*	*runtimeMinutes*	*genres*	*averageRating*	*numVotes*	*releaseDate*
tt0112159	tvSeries	Neon Genesis Evangelion	Shin seiki evangerion	0	1995	1996	24	Action, Animation, Drama	8.5	70345	20 August 1997
tt0489974	tvSeries	Carnival Row	Carnival Row	0	2019	2023	56	Crime, Drama, Fantasy	7.8	63719	30 August 2019
tt0499097	movie	Without Remorse	Without Remorse	0	2021		109	Action, Thriller, War	5.8	58538	30 April 2021
tt0800325	movie	The Dirt	The Dirt	0	2019		107	Biography, Comedy, Drama	7.0	50280	18 March 2019
tt0805647	movie	The Witches	The Witches	0	2020		106	Adventure, Comedy, Family	5.3	41745	22 October 2020

TABLE IV: Description of the full dataset

Method / Tool	*Source*	*Column*	*Dtype*	*Description*
YouTube API	trailers.csv	channelId	str	ID of the YouTube channel the trailer is posted on.
		network	str	Name of the network the channelId belongs to.
		videoId 🔑	str	ID of the YouTube video.
		videoTitle	str	Title of the YouTube video.
		publishTime	datetime	Timestamp the video was posted.
	comments.csv	videoId 🔑	str	ID of the video the comment was posted on.
		commentId 🔑	str	ID of the posted comment.
		textOriginal	str	Content of the comment.
		likeCount	int	Number of likes the comment has gotten.
		publishedAt	datetime	Timestamp the comment was published.
Return YouTube Dislike API	returnyoutubedislikes.csv	videoId 🔑	str	ID of the video the likes/dislikes were taken from
		likes	int	Number of likes the video has.
		dislikes	int	Number of dislikes the video has.
		viewCount	int	Number of views the video has.
Manual matching	match.csv	videoId 🔑	str	ID of the YouTube video
		tconst 🔑	str	IMDb ID of the entry in its database.
IMDb datasets	title.basics.tsv	tconst 🔑	str	IMDb ID of the entry in its database.
		titleType	str	Type of the entry e.g. tvSeries, movie, tvMiniSeries, tvEpisode, etc.
		primaryTitle	str	The more popular title.
		originalTitle	str	Original title, in the original language.
		isAdult	boolean	0: non-adult title; 1: adult title.
		startYear	date(yyyy)	Release year of the title.
		endYear	date(yyyy)	End year of the title.
		runtimeMinutes	int	Runtime of the title, in minutes.
		genres	str	Includes up to three genres associated with the title.
	title.ratings.tsv	tconst 🔑	str	IMDb ID of the entry in its database.
		averageRating	float	Average user rating calculated by IMDb
		numVotes	int	Number of user votes used for averageRating
IMDb scraper	release_dates.csv	tconst 🔑	str	IMDb ID of the entry in its database.
		releaseDate	date(yyyy-mm-dd)	Release date of IMDb entry, scraped from the website
Label Studio	annotations.csv	commentId 🔑	str	ID of the posted comment.
		sentimentLabel	str	Aggregate most labeled sentiment assigned to the comment.
		sentimentScore	int	The score of the sentimentLabel, negative = -1, neutral = 0, positive = 1
Data Wrangling		sentimentPredictedRaw	str	Predicted sentiment label by Vader Sentiment.
		sentimentPredictedScore	int	The score of sentimentPredictedScore, negative = -1, neutral = 0, positive = 1.
		commentDateOffset	int	Number is day(s) before/after release.

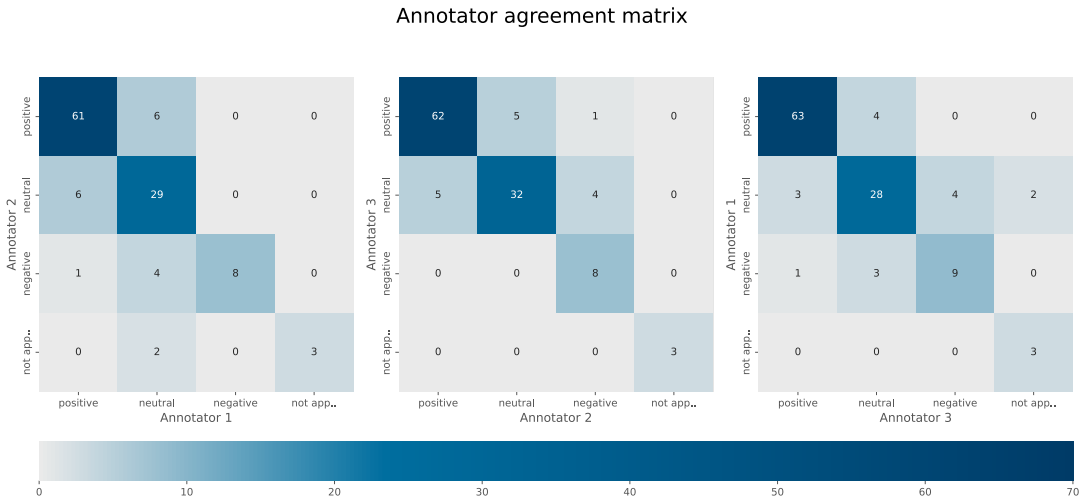

Fig. 9: 3 matrices visualizing each combination of annotators. In the diagonal, it shows the number of each class the annotators agreed on, while the non-diagonal fields show cases of disagreement.