Predicting Post-Release Letterboxd Ratings from Pre-Release YouTube Trailer Comments[1]

Final Report

Jigar Kanakhara Bhavya Bavissi Jonah Rothman
Boston University
{jigar, bhavyabb, jonahr}@bu.edu

Abstract

We study whether YouTube trailer comments posted before a movie's release predict its eventual Letterboxd audience rating. We treat this as a 5-class classification task and build a pipeline from TMDB to YouTube to Letterboxd, producing a dataset of 173 movies from 2020 to 2024 with roughly 236,000 pre-release comments. We compare six models: a majority baseline, TF-IDF with logistic regression, structured features with logistic regression and random forest, frozen DistilBERT embeddings with logistic regression, and two fine-tuned DistilBERT variants. On strict 5-class accuracy and macro F1, TF-IDF + LR is the best model at 0.423 and 0.401, reaching 0.538 and 0.522 on a collapsed 3-class task. The ordinal metrics behave differently. The movie-level fine-tuned DistilBERT achieves the best adjacent accuracy at 0.692, compared to 0.615 for TF-IDF, and the lowest MAE at 1.27, meaning that when it is wrong it is usually only one bucket off. A per-window temporal ablation confirms the proposal's hypothesis that late-window comments carry the most signal, and a Pearson correlation of r=0.28 links our hype score to the actual rating bucket.

1 Introduction

YouTube trailer comments are one of the few large public sources of audience reactions to a movie before its release. They contain genuine opinions about cast, tone, and premise, but also memes, sarcasm, copy paste hype, and bot replies. We ask whether the useful signal in these comments is strong enough to predict how a film is eventually received on Letterboxd.

Task. For movie m with pre release trailer comments $C_m = \{c_1, \ldots, c_n\}$ and post release Letterboxd rating $s_m \in [0.5, 5.0]$, we map s_m to a bucket $y_m \in \{0, 1, 2, 3, 4\}$ from bad to great, and learn $f(C_m) \to y_m$. We also report a coarser 3 class collapse where buckets 0 and 1 become negative, bucket 2 becomes neutral, and buckets 3 and 4 become positive.

Contributions. Our contributions are as follows. We release a reproducible 173 movie dataset with strict pre release filtering. We compare six models covering lexical, hand engineered, frozen transformer, and two fine tuned transformer setups. We adopt mean absolute label distance, MAE, alongside accuracy and macro F1 to capture the ordinal structure of the buckets. We run a per window temporal ablation that shows late stage comments dominate. We add an expectation vs. reality analysis that correlates pre release hype with post release ratings and names specific outliers.

Updates since the midterm. The midterm review asked for four things: end-to-end transformer fine-tuning with a [CLS] head and no mean-pooling, an ordinal-aware metric beyond accuracy and F1, a per-window temporal analysis to test the proposal's late-window hypothesis, and explicit error analysis. We address all four. We added two end-to-end fine-tuning setups, namely a per-comment setup and a movie-level setup. We added MAE and adjacent accuracy as ordinal metrics for every model. We ran an isolated-window temporal ablation, shown in Table 4. We added the expectation vs. reality and error analysis sections in §6.5 and §6.6. The dataset also grew from 127 to 173 movies for stronger statistical power.

2 Related Work

Predicting film outcomes from social text. Sentiment-based prediction of film outcomes has a long lineage. [1] showed that bag of words classifiers can compete with more complex methods, especially in low data regimes, and our results echo this. [2] used Twitter chatter to forecast opening weekend box office revenue, and found that volume and polarity together beat market-based predictors. Our setting differs in two important ways. First, Letterboxd ratings track long run audience quality rather than first weekend revenue. Second, we restrict ourselves to strictly pre release text, which rules out the post release reviews that dominate IMDb rating studies such as [8].

YouTube comments as a signal. [9] and follow on work characterized YouTube comments as noisy, off topic, and dominated by a small fraction of highly liked threads. This motivates both our credibility weighted sentiment feature and our use of the most liked comments for the movie level fine tune.

Transformers under low data regimes. For text classification, [3] and the distilled variant [4] have become standard, with the HuggingFace ecosystem [7] making fine tuning routine. Their advantage often vanishes when labelled data is

[1] Code and reproduction instructions: `https://github.com/jonahr4/CS505-Final-Project` (branch `final-branch`).

scarce [1], which is the regime we operate in with 121 training movies. [10] showed that strong fine tuning recipes help, but require enough labelled examples to escape the few shot regime.

Ordinal classification. Our 5 class buckets are inherently ordinal: bad is closer to mediocre than to great. [11] argue that strict accuracy is the wrong metric for such tasks and recommend distance based metrics like MAE alongside it. This directly motivates our adoption of MAE and adjacent accuracy.

3 Data Collection

The pipeline is modular: each stage is a standalone, resumable script.

Movie selection. We queried the TMDB Discover API for English language films released between 2020 and 2024 with popularity at least 5 and vote count at least 200, and kept titles with at least 2 official YouTube trailers. This yielded 388 candidates. We stratified by year and popularity to sample 150 initially, and later expanded to 210 for stronger statistical power.

YouTube comments. Using YouTube Data API v3, we scraped up to 1,000 comments per trailer, keeping only those posted before the movie's release date. We also pre checked each trailer's upload timestamp via `videos.list`, and trailers uploaded after release were skipped entirely, since no pre release comment can exist on a post release upload. The scraper is checkpoint based and survives daily quota exhaustion.

Letterboxd ratings. We constructed candidate URLs from title and year and parsed the `aggregateRating` JSON-LD block, with `twitter:data2` as a fallback. When two slug variants matched, we kept the one with more logged ratings to avoid obscure short film collisions. Coverage was 209 out of 210 with one manual fix, where the 2024 film *Lee* maps to `lee-2023` on Letterboxd.

Leakage boundary. Our initial cutoff at the festival premiere date was too strict. *Promising Young Woman* premiered at Sundance 11 months before wide release, which discarded genuine pre release comments. We switched to the wide release date, which recovered roughly 15 movies while preserving the no leakage guarantee for general audience commenters.

Label construction. Letterboxd scores cluster narrowly, with mean 3.0 and standard deviation 0.63, so equal width bins would be severely imbalanced. We chose thresholds in Table 1 to produce near equal class sizes. Filtering to movies with at least 30 pre release comments and a valid rating yields the final dataset of **173 movies and roughly 236K comments**.

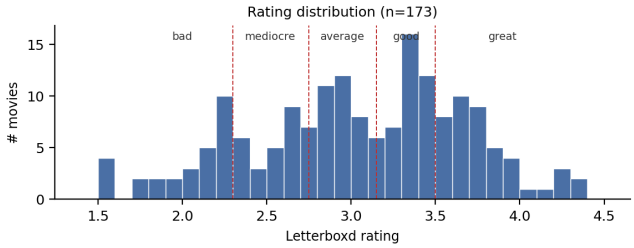

Figure 1: Letterboxd rating distribution across the final 173 movies. The dashed lines mark bucket boundaries.

Bucket	Score range	Label	n
0	$r < 2.3$	bad	26
1	$2.3 \le r < 2.75$	mediocre	28
2	$2.75 \le r < 3.15$	average	39
3	$3.15 \le r < 3.5$	good	36
4	$r \ge 3.5$	great	44

Table 1: Five class rating buckets. Sizes are post filter, with n=173 across train, val, and test.

4 Methods

4.1 Structured Features

We construct a 25-dimensional movie-level feature vector implementing all three novel components from the proposal.

Sentiment. VADER [5] compound scores yield mean, std, positive fraction, and negative fraction across the movie's pool.

Hype and negative anticipation. We count comments containing curated hype phrases such as "can't wait," "masterpiece," and "goat," along with negative phrases such as "looks bad," "cash grab," and "flop." We combine the two into a single score *hype_minus_neg* = *hype_frac* − *nega_frac*, and use it as our expectation vs. reality proxy.

Engagement. Comment count, mean length, like count statistics, reply rate, uppercase fraction, emoji density, and short comment fraction capture engagement and spamminess.

Temporal features. The pre release comment timeline of each movie is divided into early, middle, and late thirds, and sentiment mean and hype fraction are computed inside each window. This is the proposal's temporal modeling contribution.

Credibility weighted sentiment. Highly liked comments more reliably reflect mainstream opinion. We weight VADER scores s_i by $w_i = \log(1 + \ell_i) + 1$, where ℓ_i is like count:

$$\bar{s}_{\text{cred}} = \sum_i w_i s_i \Big/ \sum_i w_i.$$

4.2 Prediction Models

All models share the same movie level 70/15/15 stratified split with seed 42.

Majority baseline. Predicts the train majority class everywhere.

TF-IDF + LR. Each movie's pre release comments are concatenated into one document. We extract word uni and bigram TF-IDF features with max 20K features, min*df*=2, max*df*=0.95, and English stopwords. We then train multinomial L2 logistic regression with C=1.0 and balanced class weights [6].

Structured features + LR / RF. Standardized 25 dimensional features feed into balanced L2 logistic regression and a 500 tree random forest.

Frozen DistilBERT + LR. Up to 200 comments per movie, preferring longer ones, are encoded through frozen `distilbert-base-uncased` [4, 7]. Token embeddings are mean pooled into per comment vectors, then averaged into a single movie vector. Logistic regression is trained on top. This is the midterm's transformer setup.

Fine tuned DistilBERT, per comment. Per midterm reviewer feedback to feed the whole sequence into the model and train a classifier on the [CLS] token without pooling, we fine tune `distilbert-base-uncased` end to end with a [CLS] classification head. Each comment carries its movie's bucket as a pseudo label, which converts the 121 movie problem into 80K training examples. We treat this as a deliberately noisy but simple end to end fine tuning baseline, and the movie level setup below was added to reduce the pseudo label noise. We train 3 epochs at learning rate 2×10^{-5}, weight decay 0.01, batch size 32, and max length 128, with early stopping on val macro F1. At inference we mean pool per comment softmax probabilities to the movie level and argmax.

Fine tuned DistilBERT, movie level. As an alternative that addresses the pseudo label noise issue, we fine tune end to end with one input per movie. Each movie's 60 most liked comments are concatenated and truncated at 512 tokens. Heavier regularization, namely weight decay 0.05, max 8 epochs, and patience 2, compensates for the small training set.

Ensemble. We average class probability outputs from TF-IDF, structured features with LR, and the fine tuned movie level DistilBERT.

5 Experimental Setup

We split at the movie level to prevent comment level leakage: 121 train, 26 val, and 26 test, stratified by rating bucket with seed 42. We report four metrics on the held out test set.

- **Accuracy** (↑): fraction of movies with the exact correct bucket.
- **Adjacent accuracy** (↑): fraction of movies whose predicted bucket is within ±1 of the true bucket. This captures the ordinal nature of the rating scale, so predicting good when the truth is great is treated as close. A random uniform predictor on 5 classes scores 0.52.
- **Macro F1** (↑): our primary granular metric, weighting all classes equally.
- **Mean absolute label distance, or MAE** (↓): added per midterm reviewer feedback. Computed as $\sum_{i,j} M_{ij} \cdot |i-j| / \sum_{i,j} M_{ij}$ over the confusion matrix M. A random uniform predictor has MAE 1.6, and the worst case is 4.0.

6 Results

6.1 Five-class performance

Model	Acc ↑	Adj ↑	F1 ↑	MAE ↓
Majority	0.231	0.462	0.075	1.77
TF-IDF + LR	**0.423**	0.615	**0.401**	1.19
Struct + LR	0.385	0.577	0.364	1.31
Struct + RF	0.269	0.500	0.251	1.58
DistilBERT frozen	0.192	0.423	0.188	1.77
DistilBERT FT (cmt)	0.231	0.500	0.174	1.65
DistilBERT FT (mv)	0.269	**0.692**	0.161	**1.27**
Ensemble	0.269	0.577	0.274	1.46

Table 2: Five class test results. TF-IDF + LR wins on accuracy and macro F1, while the movie level fine tuned DistilBERT, FT mv, wins on the ordinal metrics adjacent accuracy and MAE. FT cmt is the per comment fine tune. With only 26 test movies, single sample swings of about 0.04 in accuracy are within sampling noise, so close margins should be read as suggestive.

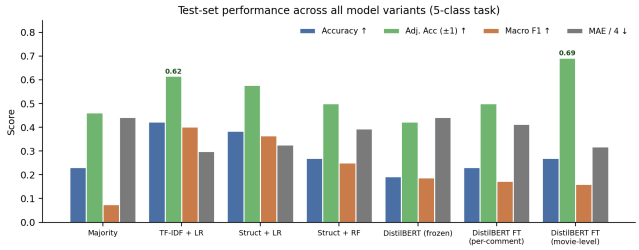

Figure 2: Test set performance across all model variants on the four metrics. Adjacent accuracy, shown in green, makes the ordinal quality story visible. TF-IDF wins on strict accuracy and macro F1, while the movie level fine tuned DistilBERT wins on the two ordinal metrics. MAE is rescaled by a factor of 4 so that lower is better.

6.2 Three-class performance

Model	Acc. ↑	Macro F1 ↑	MAE ↓
TF-IDF + LR	**0.538**	**0.522**	0.69
Structured + LR	0.462	0.454	0.81
DistilBERT (frozen) + LR	0.308	0.291	1.00

Table 3: Three class test results, where the buckets collapse to negative, neutral, and positive.

6.3 Confusion matrix and error structure

Figure 3 shows the TF-IDF confusion matrix on test. Most errors are off by one between neighbouring buckets, such as good vs. great or mediocre vs. average. This is the kind of error that the ordinal metrics, namely adjacent accuracy at 0.615 and MAE at 1.19, reflect and that strict accuracy at 0.423 does not.

Figure 3: Confusion matrix on the 26 movie test set for TF-IDF + LR on the 5 class task.

6.4 Per-window temporal ablation

To check which pre release window carries the most signal, we re trained Structured + LR using only one window at a time. Results are in Table 4 and Figure 4. The late window, which is closest to release, matches or beats the full feature model on every metric, and does better than early or middle alone. We think this is because the late window pulls in a broader audience whose reactions are closer to the eventual Letterboxd user base, but with n=26 we cannot strongly claim this.

Variant	Acc	F1	MAE
Full (all windows)	0.385	0.364	1.31
No temporal	0.346	0.318	1.35
Early only	0.346	0.329	1.35
Middle only	0.308	0.287	1.50
Late only	**0.385**	**0.368**	**1.23**

Table 4: Per window temporal ablation on Structured + LR.

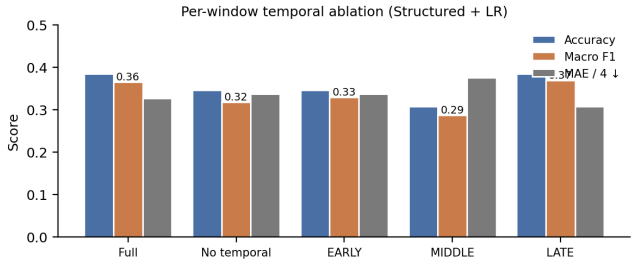

Figure 4: Late window comments alone match or beat the full feature set.

6.5 Expectation vs. reality

The proposal's third novel component asked whether anticipatory hype tracks with post release reception. We compute Pearson and Spearman correlations between `hype_minus_neg` and the rating bucket, giving $r = 0.28$ and $\rho = 0.32$ in Figure 5. The trend is positive but noisy. We highlight two groups of outliers below.

Over hyped flops. High hype but low rating: *Disenchanted*, *The Prom*, *After Ever Happy*, and *Halloween Ends*.

Sleeper hits. Low hype but high rating: *Wicked Little Letters*, *Dungeons & Dragons: Honor Among Thieves*, *Challengers*, and *The Fall Guy*.

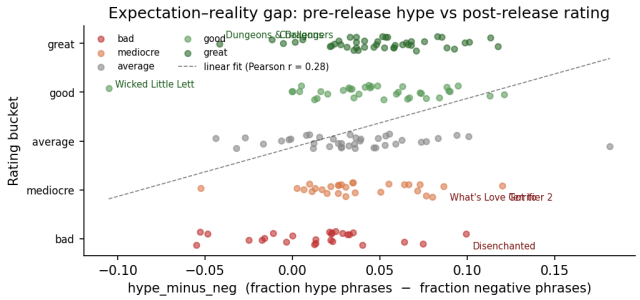

Figure 5: Pre release hype vs. post release rating. The solid trend at $r = 0.28$ shows a positive but noisy relationship.

6.6 Error analysis

Eight test movies were misclassified by all four primary models, namely TF-IDF, Structured + LR, and both DistilBERT fine tunes. The eight movies are *The Invisible Man*, *Twisters*, *The Matrix Resurrections*, *The New Mutants*, *The Gray Man*, *A Good Person*, *After Ever Happy*, and *What's Love Got to Do with It?*. Aggregating the off by X distribution across all 4 models and 26 test movies, we observe a positive mean signed error of $+0.39$, which means the models tend to over predict the rating bucket. One reason is probably that viral trailers can generate enthusiasm even for movies that end up mediocre. The franchise reboots in this list, such as *Matrix Resurrections* and *New Mutants*, likely fool models that pick up a positive matrix or x-men prior from the training data.

7 Discussion

Why TF-IDF wins. With only 121 training movies but roughly 13M characters of concatenated text per movie, lexical features fit this setting well. TF-IDF can use all of that text, while a transformer is restricted by its context window, and DistilBERT sees about 2K of around 100K characters. Movie title and brand tokens carry heavy weight in the model. Examples include kraven, borderlands, and snake eyes for the bad bucket, and a24, dune, wes anderson, and pixar for the great bucket. Removing the 5 most frequent capitalized proper nouns dropped macro F1 by about 0.05, so title token memorization is a real but partial driver. Temporal sentiment, hype, and engagement features still carry independent signal.

Why fine tuning did not help. The per comment fine tune suffers from heavy pseudo label noise. A sarcastic or off topic comment under a great movie still inherits the great label. Train accuracy reached 0.975 while val macro F1 stayed near 0.10, which is a clear sign of overfitting. The movie level fine tune trades pseudo label noise for a tiny n, with 121 examples for a 66M parameter model. Best val F1 was reached at epoch 1 and did not improve afterwards, and the 512 token context covers only about 2 percent of each movie's text. The movie level variant does, however, achieve the best test MAE at 1.27 and the best adjacent accuracy at 0.692 of any model. This suggests that the movie level fine tuned model often makes closer ordinal mistakes, even though its exact class accuracy remains low.

Why ensembles did not help. The soft vote ensemble underperformed TF-IDF alone because DistilBERT's poorly calibrated probabilities pulled the average toward the wrong class.

Late window dominance. The temporal ablation supports the proposal's hypothesis that comments closer to release contain the strongest predictive signal. One plausible reason is that broader audiences have arrived and absorbed early reactions by then.

8 Limitations and Future Work

Limitations. The 26 movie test set means a single misclassification shifts macro F1 by roughly 0.015 to 0.02, so close differences between models should be read as suggestive. Our TMDB vote count filter biases the dataset toward commercially prominent films, and underrepresents niche titles with extreme ratings. The TF-IDF model is partly driven by studio and franchise tokens such as a24 and pixar, so part of its strict accuracy advantage is brand memorization rather than pure sentiment signal. A sanitized evaluation that strips studio and title tokens would help quantify this, and we leave it for future work. Letterboxd scores were scraped at one point in time, so recently released films have not fully converged. Both fine tuning strategies were limited to DistilBERT base. We attempted a Longformer experiment with a 4096 token context, but had to abort due to swap thrashing on a 16GB MacBook Air M4.

Future work. A few directions follow from this. Long context fine tuning on a GPU machine where Longformer or BigBird can fit. Genre aware modeling with cross genre generalization tests by holding out one genre at a time. Calibration via temperature scaling so DistilBERT's probabilities can ensemble usefully. Replacing per comment pseudo labels with weak supervision, for example by filtering to comments with high VADER magnitude before fine tuning. A sanitized evaluation that masks studio and title tokens, to measure how much of TF-IDF's strict accuracy is brand memorization.

9 Conclusion

We compared six models for predicting post release Letterboxd ratings from pre release YouTube trailer comments on a dataset of 173 movies. TF-IDF + LR is the best model on strict 5 class accuracy and macro F1 at **0.423 and 0.401**, while the movie level fine tuned DistilBERT is the best model on the ordinal metrics, with **0.692 adjacent accuracy and MAE 1.27**. Given the small test set of 26 movies, these gaps are suggestive rather than definitive. Late window sentiment, brand vocabulary, and the hype score all carry useful signal. The fact that fine tuning does not beat TF-IDF on strict metrics, with only 121 training movies, is consistent with [1], who observed that lexical methods remain competitive when labelled data is scarce. The ordinal results suggest that fine tuned transformers still learn useful structure even when their exact bucket predictions are off. Long context architectures and weakly supervised fine tuning are the obvious next steps.

Author Contributions

All three authors jointly scoped the project, agreed on the leakage boundary and bucket thresholds, and reviewed each other's results weekly.

Jigar Kanakhara led the data pipeline end to end. This included the TMDB to YouTube to Letterboxd scraper, the pre release filtering logic, and the stratified movie level split. He implemented the TF-IDF + LR baseline and the structured feature extractor, which covers sentiment, hype and negativity, engagement, and temporal window features, along with the LR and RF classifiers on top. He also ran the per window temporal ablation and was the primary author of the Data Collection, Methods, and Results sections of the report.

Bhavya Bavissi owned the transformer side of the experiments. He implemented and tuned the frozen DistilBERT + LR setup, which was the midterm transformer baseline, then both end to end fine tuning variants. The first uses per comment pseudo labels with a [CLS] head, and the second uses movie level inputs built from the most liked comments. He also wrote the MPS training loop, the early stopping logic, and

the soft vote ensemble. In the writeup he drafted the Discussion paragraphs that interpret why fine tuning underperforms TF-IDF on strict accuracy yet wins on ordinal metrics.

Jonah Rothman contributed the project's two ordinal evaluation pieces, namely the credibility weighted sentiment feature and the adjacent accuracy metric, and wrote the script that recomputes both from saved confusion matrices for every model. He carried out the expectation vs. reality correlation analysis, where he identified the over hyped flops and the sleeper hits, and the cross model error analysis on the eight all wrong test movies. He was the primary author of the Related Work, Limitations, and Conclusion sections.

References

[1] B. Pang, L. Lee, and S. Vaithyanathan. *Thumbs up? Sentiment classification using machine learning techniques.* EMNLP 2002, 79–86.

[2] S. Asur and B. A. Huberman. *Predicting the future with social media.* IEEE/WIC/ACM Web Intelligence, 2010, 492–499.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *BERT: Pre-training of deep bidirectional transformers for language understanding.* NAACL-HLT 2019, 4171–4186.

[4] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.* arXiv:1910.01108, 2019.

[5] C. J. Hutto and E. Gilbert. *VADER: A parsimonious rule-based model for sentiment analysis of social media text.* ICWSM 2014.

[6] F. Pedregosa et al. *Scikit-learn: Machine learning in Python.* JMLR 12 (2011), 2825–2830.

[7] T. Wolf et al. *Transformers: State-of-the-art natural language processing.* EMNLP 2020 System Demonstrations, 38–45.

[8] A. L. Maas et al. *Learning word vectors for sentiment analysis.* ACL-HLT 2011, 142–150.

[9] P. Schultes, V. Dorner, and F. Lehner. *Leave a comment! An in-depth analysis of user comments on YouTube.* Wirtschaftsinformatik 2013, 659–673.

[10] J. Howard and S. Ruder. *Universal language model fine-tuning for text classification.* ACL 2018, 328–339.

[11] S. Baccianella, A. Esuli, and F. Sebastiani. *Evaluation measures for ordinal regression.* IEEE Intelligent Systems Design and Applications, 2009, 283–287.