Sentiment Analysis on Movie Reviews [Mini]

Maël Dieudonné
ENSAE
mael.dieudonne@ensae.fr

Abstract

Sentiment analysis aims to capture the subjectivity expressed in written texts. It is most commonly performed at the document level using a binary approach—classifying the overall emotional tone as either positive or negative. This method is frequently applied to movie reviews, which tend to be longer and more readily available than other subjective texts, such as product reviews.

The dataset in question, published in 2011, consists of 50,000 IMDb movie reviews labeled as either positive or negative based on their associated rating (respectively 7 or higher and 4 or lower; reviews associated with ratings of 5 and 6 were deliberately excluded).

Current SOTA models on this dataset are fine-tuned encoders, but their performance seems to have reached a plateau. Replicating these models would require excessive computational resources. Instead, I have chosen to investigate an alternative explanation: that this plateauing is due to limitations within the dataset itself.

I compared the performance of four models: three based on RoBERTa (the base model for zero-shot classification, my own fine-tuned version trained on the IMDb dataset, and a SOTA model fine-tuned specifically for sentiment analysis) and GPT-3.5 Turbo.

In selecting these models, I explored two key factors: the impact of prompt design on zero-shot classification, and the influence of classifier head architecture on fine-tuning performance. The results show that while prompt engineering can significantly boost performance, modifying the classifier head yields comparatively limited gains.

All models exhibit a significant drop in accuracy for reviews with ratings of 4 and 7. Combined with a cursory examination of the dataset, this suggests that ratings may not be an optimal proxy for the sentiment expressed in movie reviews. If so, this could introduce noise into the true labels, making perfect prediction inherently unattainable.

1 Introduction

Sentiment analysis seeks to understand the subjectivity expressed in written documents. Opinion and views are often used interchangeably with sentiment in the literature, in contrast to factual statements [1]. Sentiment analysis has become an active research area with the advent of Web 2.0, offering people numerous opportunities to express themselves in a publicly traceable manner [2]. Various stakeholders have a vested interest in their opinions:

- **Marketing departments**, seeking to understand brand perception and consumer opinions about products, both in absolute terms and relative to competitors.
- **Customer service teams**, identifying key factors contributing to customer satisfaction or dissatisfaction.
- **Traders and financial analysts**, using public sentiment as an early indicator of economic trends and the financial health of companies.

- **Political actors**, monitoring public perception of policies and personal image.
- **Public health officials**, assessing attitudes toward medical interventions such as vaccines.
- Etc.

The primary goal of sentiment analysis is to gain direct access to people's opinions without relying on intermediaries such as journalists or resellers, at the massive scale typical of the Internet [3, 4].

Sentiment analysis can be performed at three distinct levels. At the **document level**, the goal is to determine the overall sentiment of an entire document, which can be very short (e.g., product reviews). Analysis conducted at the **sentence level** account for potentially conflicting sentiments within the same documents. First, it distinguishes between objective (fact-based) and subjective (opinion-based) sentences. For example, a movie review may contain both plot summaries (objective) and judgments on acting performance (subjective). Then, subjective sentences are classified based on their sentiment. At the **feature** or **aspect level**, the aim is to link sentiments with their object, and sometimes the person expressing them (who feels what about the plot, for instance).

The most common output of sentiment analysis is a binary or ternary classification of the emotional tone, also known as polarity or valence: positive, negative or neutral. This classification applies to entire documents, individual subjective sentences, or specific features, depending on the level of analysis. Sentiment classification beyond polarity, aiming at identifying specific emotions like fear, anger, or happiness, is less common. This approach may have fewer applications outside of literary analysis.

Sentiment analysis faces several important challenges [5]:

- **Domain dependance**, when words carry different meanings in different contexts (e.g., tears may indicate laughter or sadness).
- **Polarity shift**, when words or punctuation marks change the valence of a sentiment (e.g., not, but, only).
- **Sarcasm**, when the expressed sentiments oppose the actual sentiments.
- **Order dependance**, when the valence of a word changes with its position in the sentence (e.g., with comparisons).
- **Unconventional expressions** like idioms, slang, emojis, or multiple punctuations marks. Implicit sentiment, when sentiments are expressed without explicit sentiment words (e.g., "I know real Indian food and this wasn't it").

2 State of the art

Three main approaches have been applied to sentiment analysis, making use of lexicon-based models, classifiers, and transformers [6].

2.1 Lexicon-based

This approach relies on a dictionary assigning positive or negative labels to words, to classify documents based on the most frequent label. A well-known example is VADER, which was developed to analyze the sentiments of reviews published on social media [7]. It relies on sentiment lexicon of 7,520 words including slang, emoticons, and acronyms, where each word is associated with a score on a valence scale (e.g., -2.2 for ":(", 0.9 for "okay" or 3.1 for "good"). This lexicon was obtained by submitting a list of 9,000 words to human raters recruited on the Amazon Mechanical Turk. Each word was rated by 10 different persons on a 9-point Likert scale, and kept in the lexicon if its average rating was non-null. VADER also applies rules that adjust sentiment scores based on punctuation, capitalization, and specific words (e.g., "but" shifts sentiment, while "!!" amplifies intensity). Finally, VADER calculates the proportion of text conveying positive, negative, or neutral sentiment.

This approach has several limits. Lexicons are complex to build, as they require a lot of human input, so they tend to remain small and not updated very often. They also lack flexibility, being unable to handle words they do not include. This can be especially problematic when processing content drawn from the Internet, which may include words written incorrectly or with evolving meaning. Since the

analysis occurs at the word level, it lacks contextual understanding. As a result, the performance of lexicon-based sentiment analysis is rather low.

2.2 Classifiers

A second approach consists of using standard machine learning algorithms. Almost any classification algorithm can be applied; the challenge lies in preprocessing the text to make it suitable for the classifier, typically through a bag-of-words representation. This method produces a high-dimensional but sparse feature space, making regularization techniques particularly useful. These techniques reduce the vocabulary—similar to a lexicon—but do so based on a word's effectiveness in sentiment prediction rather than a predefined list. An example is available here with support vector machines, random forest and neural networks [8].

The use of classifiers can be seen as a generalization of lexicon-based approaches. Unlike lexicon-based methods, classifiers do not require a predefined dictionary specifying which words to consider and how to interpret them. Instead, models autonomously select the words they rely on for classification, along with their associated sentiment (i.e., emotional tone). They enable more complex mappings between the input (i.e., document) and the output (i.e., predicted sentiment) beyond simple relative word frequencies. Classifiers can also incorporate external features, such as a reviewer's personal characteristics, to refine predictions.

However, classifiers have limitations. They require labeled datasets to train the models, though these datasets are generally easier to construct than sentiment dictionaries. Despite their advantages, classifiers may still struggle with understanding connotations and context, as they analyze words in isolation. While they perform well on short documents, they often have difficulty handling longer texts that express conflicting viewpoints.

2.3 Transformers

The third approach uses transformer models, which apply attention mechanisms to word embeddings—low-dimensional, dense representations of text. There are two main types of models sharing the same basic architecture:

- **Encoder/decoder** (a.k.a. unidirectional or autoregressive transformers): these models consider only past and present tokens, with the aim of predicting future tokens (which are removed from through causal masking). They are easily trained by comparing predicted and actual tokens. The reference architecture for encoders/decoders is GPT [9, 10].

- **Encoder-only** (a.k.a. bidirectional transformers): these models consider all tokens in the context window, with the aim of understanding text. They are trained by masking a fraction of input tokens (usually 15%) before feeding them to the model, then predicting the masked tokens. Additional training strategies include predicting the order of two sequences in the training corpus or generating permutations of tokens within sentences. Lacking decoders, these models output embeddings instead of tokens. These embeddings can be analyzed directly, e.g., through cosine similarity to identify semantically similar tokens, which is essentially the principle of **zero-shot classification**. Alternatively, a custom output head can be added and fine-tuned for a specific task. The reference architecture of encoders is BERT [11].

Large language models offer powerful tools for sentiment analysis, with a unique ability to capture context and disentangle sentiment from other features. They excel in complex tasks, particularly aspect-based sentiment analysis. However, they have several limitations. They are resource-intensive, especially during pre-training. As a result, the most common approach is to use pre-trained encoders models, either directly or for fine-tuning. A less frequent approach is to run GPT models locally or access them via APIs. Another challenge with transformers is their opacity; what makes them so effective remains difficult to assess. Lastly, they may not outperform traditional classifiers on short documents [12].

3 Data

Movie reviews have become a popular benchmark for sentiment analysis due to their availability and size: they tend to be longer and more accessible online compared to typical product reviews or social media posts. The current dataset was constructed and publicized by Maas et al. [13]. It contains 50,000 reviews from the IMDb website, along with the corresponding rating proposed by theirs authors on a 1-10 scale. Reviews are labeled as negative when associated with ratings below 5, as positive when associated with ratings above 6. Neutral ratings (5 and 6) were eliminated to focus on polarized reviews. To minimize bias, no more than 30 reviews per movie were included, and an equal number of positive and negative reviews were selected, ensuring that random guessing would yield 50% accuracy. The dataset is split into training and test samples of equal size (25,000 reviews each) [1] .

The task, therefore, is to perform binary classification between positive and negative sentiments at the document level.

3.1 Benchmarks

Performances are assessed regarding accuracy, which is the most relevant metrics for a binary classification problem where each class is of equal importance and the dataset is balanced. This is different from, e.g., diagnosing a serious but rare disease, where false positives are preferable to false negatives, and recall would be the performance metric of choice.

Several models were benchmarked on the IMDb dataset, reaching the following error rates on the test sample [2]:

- 26.4% for a **lexicon-based model**,
- 12.3% for a **Lasso logistic regression**,
- 5.91% for an **adversarial neural network**.

The current best model (according to the Papers with Code leaderboard) is a **fine-tuned BERT** model named LlamBERT (Csanády et al., 2024) with an error rate of 3.32%. It is followed closely by an **auto-regressive transformer** named XLNet (Yang et al., 2019) with an error rate of 3.79%.

Transformers are significantly outperforming lexicon-based models and traditional classification algorithms. This is not surprising, as they leverage much more information, being trained on corpora orders of magnitude larger than the IMDb dataset. However, they seem to have reached a plateau, with only marginal improvements over the last 6 years. Additionally, they are difficult to replicate due to heavy hardware requirements. It seems clearly out of my reach to improve upon these models.

4 Experiment

4.1 Hypotheses

Instead, I have set out to investigate the possibility that performance is plateauing due to limitations of the dataset, with two hypotheses in mind.

(1) The way reviews are labeled means that the prediction task is not based on the sentiment conveyed by reviews, but on the associated ratings (below 5 or above 6). However, these ratings may not always be a reliable proxy for the overall sentiment of reviews, especially when they are not strongly polarized. Below are a few examples of reviews labeled as positive, despite their tone being quite ambiguous:

> Definitely a movie that could've been seen on DVD. (rating 7)
>
> Moviegoers wanting an inside look at what it's like to embark on a daring rescue mission in the middle of the ocean might want to give The Guardian a chance. I saw it for free, but had I paid I would've felt I had gotten my money's worth. (rating 8)

[1] The dataset also includes an additional 50,000 unlabeled reviews without ratings. These can be used for unsupervised learning but not to measure performance.

> In the end, "The Dresser" will bore many people silly, but it will truly be a delight to those who love British cinema. (rating 8)
>
> The theme is controversial and the depiction of the hypocritical and sexually starved india is excellent.Nothing more to this film.There is a lack of good dialogues(why was the movie in english??). There was lack of continuity and lack of passion/emotion in the acting. (rating8)

The working hypothesis here is that the error rate is higher for models closer to neutral, especially with a rating of 7.

(2) The document-level approach is not ideal, as many reviews express conflicting opinions about different aspects of a movie, making it difficult to assign a single, reliable sentiment.

> Some things and some actions are very true, but some other stuff is just way off the mark. (...) Delhi is complicated. India is complicated. The director tries to simplify both. And fails pretty miserably at that. (...) But the chemistry between the leads is palpable. (rating 8)
>
> The plot is rather thin, yet it does have a moral. It's about the perils of quick fame and fast money, and how any happiness with only these factors will ultimately be doomed. This will cheer you up, lift you up to your feet and make you laugh. (...) Another major problem is that too many songs are just not that great. (...) The grandmother sub-plot borders on being ridiculous. (rating 8)

The working hypothesis here is the lengthier the reviews, the more likely they are to express contrasted opinions, and the harder they become to classify.

4.2 Models

My goal was to compare several models to assess their behavior in relation to ratings and review lengths, while striving to get as close as possible to the SOTA performance. I chose RoBERTa Large [14] as my base model , since the SOTA model is built upon it and it also performs really well on its own (outperforming XLNet, which serves as the basis for the second SOTA model) [2] .

RoBERTa's architecture is largely similar to the original BERT model, with only minor tweaks to some hyperparameters. It has a vocabulary size of 50,265, a hidden size of 768 for RoBERTa base and 1024 for RoBERTa large, 12 attention heads with 12 layers each and an intermediate size of 3072. The context length is 512 tokens. The sole difference between BERT and RoBERTa architectures regards tokenization. RoBERTa uses Byte-Pair Encoding, which operates at the subword level with an agglomerative approach, merging the most frequent pairs iteratively until the desired vocabulary size is reached. Additionally, RoBERTa omits `token_type_ids`, which are used in BERT to distinguish between prompts and answers.

Where BERT and RoBERTa truly differ is in their training procedures. RoBERTa eliminates the next sentence prediction objective and employs dynamic masking, meaning that input tokens are masked differently every 4 epochs rather than being fixed during preprocessing. Additionally, RoBERTa's training algorithm uses larger mini-batches and significantly fewer training steps (30,000 compared to 1 million for BERT). Its training corpus is also substantially larger, at approximately 160 GB. While I could not confirm whether it includes the IMDb dataset, it does use OpenWebText, which is built from content extracted from URLs shared on Reddit with at least 3 upvotes.

I use zero-shot classification with the base RoBERTa model through a simple HuggingFace pipeline as a baseline for performance, comparing it with the following models.

4.2.1 Fine-tuned RoBERTa on the IMDb dataset

The standard fine-tuning approach freezes the base model's parameters (354,315,266 for RoBERTa-large) to preserve the knowledge gained during pretraining and reduce computational overhead. This setup allows for an elegant optimization: the dataset only needs to pass through the base model once to generate embeddings, after which a classifier can be trained directly on them, eliminating the need

[2] It could have been interesting to use ModernBERT [15] instead, which is more recent, but I discovered its existence too late.

for full forward passes at each epoch. Surprisingly, despite extensive reading on fine-tuning of BERT models, I never saw that trick mentioned. Yet, it is a game changer. In my case, this reduced the computation time per epoch from 1 hour to just 1 second, revealing that 99.97% of the computation time was previously wasted regenerating embeddings rather than actually training the classifier. Indeed, the task shifts from tuning an LLM to simply fitting a classifier on fixed embeddings: the LLM complexity effectively disappears.

I used this trick to compare several classification heads of increasing complexity—a topic I was also surprised to discover as underexplored. The configurations I tested were the following.

1. **Two dense layers**, the first mapping the model output to a hidden space of the same (embedding) dimension, the second projecting back to a binary space (this corresponds to the default head RoBERTa-large, totaling $(1024+1)*1024+(1024+1)*2 = 1,051,650$ trainable parameters).
2. **Three dense layers**, the first expanding the model output to a hidden space twice as large (2048), the second reducing it to an intermediate dimension (512), and the last projecting to a binary space (making for $(1024+1)*2048+(2048+1)*512+(512+1)*2 = 3,149,314$ trainable parameters).
3. With a SiLU **activation layer** after the first dense layer.
4. With a **normalization layer** before the activation layer (adding $2048 * 2 = 4096$ trainable parameters for a total of $3,149,314$).
5. With 2 **dropout layers**, one placed before the normalization layer and the other before the last dense layer, both using a drop-out rate of .
6. With a **residual connection** after the activation layer.

These classification heads were applied to the `[CLS]` token, which marks the beginning of the input sequences and serves as an embedding for their entirety [3] .

The initial embedding was performed with a `batch_size` of 250. The models were trained during 300 epochs. A relatively high learning rate of 10^-3 was used in combination with a cosine learning rate scheduler, featuring a 10% warm-up phase. This setup was designed to ensure stable convergence and to maximize the performance potential of each model.

4.2.2 SOTA fine-tuned RoBERTa for sentiment analysis

To assess the effectiveness of my fine-tuning, I compared it not only to the baseline performance but also to the current SOTA fine-tuned RoBERTa for sentiment analysis, which is SiEBERT [6]. Its architecture is not specified, as the authors do not describe it and have not released their code. It is unclear whether SiEBERT uses a custom classification head or the default untrained head provided by RoBERTa, which consists of two dense layers (the first mapping to a hidden space of the same size as the RoBERTa output, the second to a binary space). The training corpus is also unspecified; the authors only mention using all publicly available two-class sentiment datasets they could identify, excluding their training subset.

SiEBERT is available in a HuggingFace pipeline, which I used.

4.2.3 GPT 3.5 turbo

I decided to add a comparison with ChatGPT, which has shown promising results for sentiment analysis but has never been applied to the IMDb dataset [8]. I drafted a first prompt based on the principle of few-shot prompting, which consists in including a few examples before the document to classify. This has been observed to improve the performance of sentiment analysis significantly—an effect known as in-context learning [16]. I then submitted this prompt to ChatGPT, asking for improvements, and received the suggestion of adding detailed instructions. Prompts were constructed as follows for each review of the test sample:

> Here is an example of a positive movie review, associated with a rating of 10:

[3] I have tried applying it to the whole input sequence, pooling tokens through their mean or max, on a subset of reviews: the performance were worse.

*[<*random review with a positive sentiment from the training set*>]*

Here is an example of a negative movie review, associated with a rating of 2:

*[<*random review with a negative sentiment from the training set*>]*

Now, consider the following review. Based on its content, is the sentiment of the review positive or negative? Answer with a single word.

*[<*review to classify from the test sample*>]*

Instructions:

- **Positive** reviews typically highlight enjoyment, satisfaction, or praise for aspects of the film (e.g., acting, storyline, direction).

- **Negative** reviews tend to criticize the film for its shortcomings or failures (e.g., poor pacing, bad acting, or unsatisfying plot).

- Focus on the general tone of the review as a whole, not isolated statements or minor contradictions.

I queried ChatGPT through the OpenAI API. I chose GPT 3.5 Turbo as the best compromise between cost and performance. At $0.5 per million input tokens and $1.5 per million output tokens, with 25,582,485 input tokens from building the prompt and 25,000 expected output tokens, the estimated cost was $12.83.

An inherent challenge with this approach is data leakage. There is a high likelihood that the IMDb test sample was included in GPT-3.5 Turbo's training corpus, which could lead to overestimated performance, as it may not be working with truly unseen data. The solution would have been to build a new dataset by scrapping reviews published after the cutoff date of GPT 3.5 Turbo, which is September 2021, but this would have added significant complexity to the project. Instead, I assumed that the IMDb dataset is small enough to be diluted in the larger training corpus of GPT-3.5 Turbo, making significant memorization unlikely.

4.3 Data preprocessing

Text was cleaned consistently across all models to ensure comparability of their results:

- Removing HTML markups, primarily line breaks (replacing `<br>` with `\n` to preserve the paragraph structure).
- Removing a few emojis that are not linked to sentiment (® and ©).
- Replacing hexadecimal characters by their ASCII equivalent.

5 Results

5.1 Data

Two characteristics of the dataset are not mentioned in the documentation but nonetheless crucial.

1. A cursory examination suggests that movies are different in each dataset, even if they share the same id. This is excellent for preventing data leakage between train and test samples.
2. All reviews goes in pair, with each movie having exactly one positive and one negative review. This forbids the models from learning non-sentiment-related features (e.g., related to the plot of a movie having overwhelmingly positive reviews in the dataset).

The reviews are highly polarized, with nearly half of the dataset consisting of ratings of 1 or 10. The distribution of ratings is almost identical in both the train and test samples. Most reviews are relatively short, but there is a long tail in the distribution of review lengths. Some reviews are exceedingly long, with a maximum of 2,459 words (for an average of 231 words in the train sample, 223 in the test sample). Review length and ratings are almost uncorrelated, with a Pearson correlation coefficient of 0.0148. This indicates that my two hypotheses can be tested independently.

Candidate Labels	Accuracy
positive sentiment / negative sentiment	0.83056
positive / negative	0.85908
favorable opinion / unfavorable opinion	0.87596
positive review / negative review	0.89136
excellent / terrible	0.91140
good movie / bad movie	0.91708

Table 1: Performance of various candidate labels for zero-shot classification with RoBERTa

Model	Left side	Right size
RoBERTa	0.9171	0.9160
SiEBERT	0.9573	0.9559

Table 2: Effects of truncation side on performance

5.2 Candidate Labels for Zero-Shot classification with RoBERTa

The initial run with "positive" and "negative" as candidate labels yielded relatively low accuracy. To improve performance, I experimented with more informative candidate labels. The best results were achieved using "good movie" and "bad movie" as labels, which resulted in an improvement of nearly 7 percentage points in accuracy compared to the "positive" and "negative" labels (see Table 1).

5.3 Truncation

RoBERTa models have a context length of 512 tokens, but many reviews are longer than that. The easiest solution is to truncate them, even though that means loosing part of the information on their overall sentiment. This truncation can be made from the right, keeping the end beginning of sequences, or from the left, keeping the end of input sequences. This could have significant consequences depending on the order in which information is presented in the reviews (e.g., if users always start with positive aspects, and always end with negative aspects).

Left-side truncation yielded marginally better results (using "good movie" and "bad movie" as candidate labels for RoBERTa, see Table 2). This could be explained by the tendency of some users to summarize their sentiment at the end of their reviews: such summaries would be lost with right-truncation, erasing important information. Although the overall effect is really tenuous, the left-hand side truncation was retained for the fine-tuned RoBERTa model. It was also used in the previous comparison between candidate labels.

5.4 Fine-tuning

It was done on a MacBook Pro with an M2 Max equipped with 38 GPU cores and 64 GB of shared memory, using the MPS backend of PyTorch, whose performance significantly outpaced the 16 GB Tesla T4 available in the Datalab or Google Colab.

The effect of training is notable for accuracy and classification error [4] (see Figure 1). Overall, all models perform similarly, with differences in accuracy appearing only in the third decimal place (see Table 3). The best model is only 0.467% more accurate than the worst model. There seems to be a clear performance ceiling at around 0.938 accuracy. No matter how many runs or specifications I tried—including exploratory tests not presented here regarding hidden size and dropout rates—I could never exceed this value. This suggests that the model's ability to learn beyond pre-training is limited, at least with the small sample I have here.

[4] Which reflects the separation between predicted classes, that is, the confidence of the model in its own predictions. Classification error is defined as the average difference between the highest possible probability ($= 1$) and the highest predicted probability of each datapoint (so the more certain are the label assignments, the closer E gets to 0): $E = \frac{1}{N} \sum_{i=1}^{N} (1 - P_{\max,i})$.

Model type	Parameters	Best epoch	Accuracy	Classification error
1	1,051,650	297	0.9366	0.0649
2	3,149,314	298	0.9374	0.0644
3	3,149,314	294	0.9379	0.0608
4	3,153,410	72	0.9334	0.0609
5	3,153,410	74	0.9338	0.0634
6	3,153,410	129	0.9353	0.0513

Table 3: Performance of Fine-tuned RoBERTa models

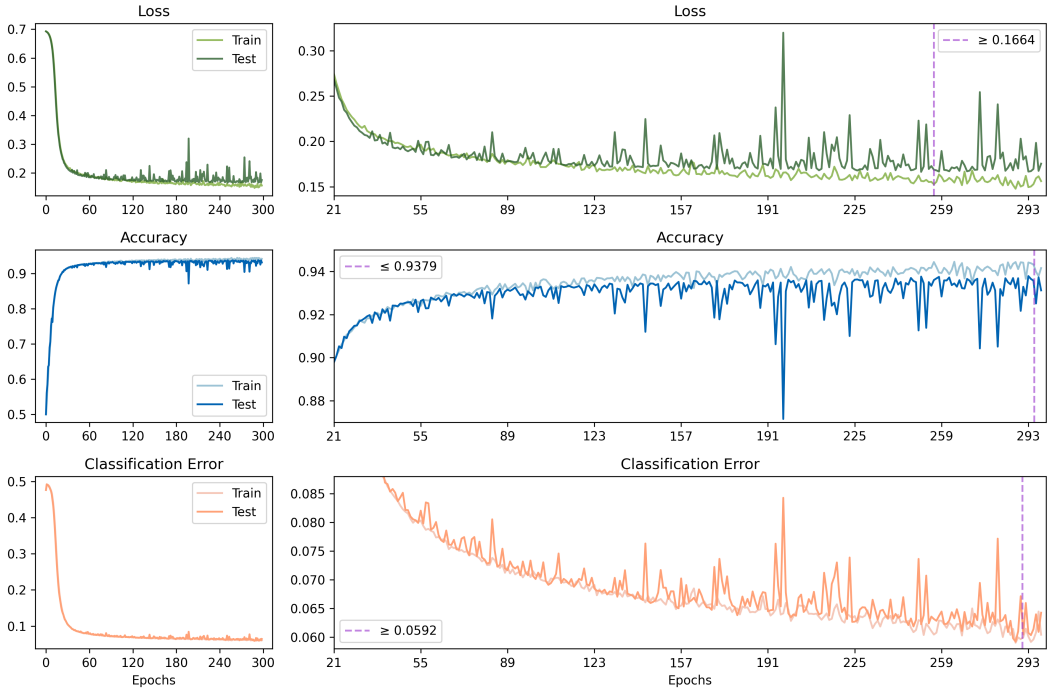

Figure 1: Training Performance of Model 3

Although the architecture of the classifier head has limited influence on performance, it still affects the results. The three most complex models are the quickest to converge but also the least performant. The best performing model is relatively simple, with three dense layers and one activation layer. It shows a notable improvement of 2 percentage points in accuracy compared to the best-performing zero-shot labels with the base models, and a smaller improvement of 0.0007 percentage points compared to the default classification head.

5.5 Performances

All models achieve over 90% accuracy, but none approaches the SOTA performance of 96.68% (see Table 4). Zero-shot classification with RoBERTa Large and optimized prompting performs the worst—unsurprisingly, as it is the most generic model. The performance of GPT-3.5 Turbo falls between the two RoBERTa models, highlighting that despite its capabilities and the use of advanced prompting techniques, specialized encoders can still outperform it [5] . The best-performing models are fine-tuned for sentiment analysis. The difference between my fine-tuned RoBERTa and SiEBERT reflects the fact that SiEBERT was fine-tuned on a much larger dataset than IMDb reviews.

[5]It would have been valuable to explore alternative prompt engineering strategies and to test other GPT models. However, this was impractical due to the cost and the API's slow speed—averaging 8 seconds per call, which would have resulted in over 55 hours to process the entire dataset.

Model	Sentiment		Average
	Negative	Positive	
RoBERTa base	0.8710	0.9631	0.9171
GPT	0.9623	0.8978	0.9300
RoBERTa fine-tuned	0.9346	0.9411	0.9379
SiEBERT	0.9520	0.9626	0.9573

Table 4: Performances of each model

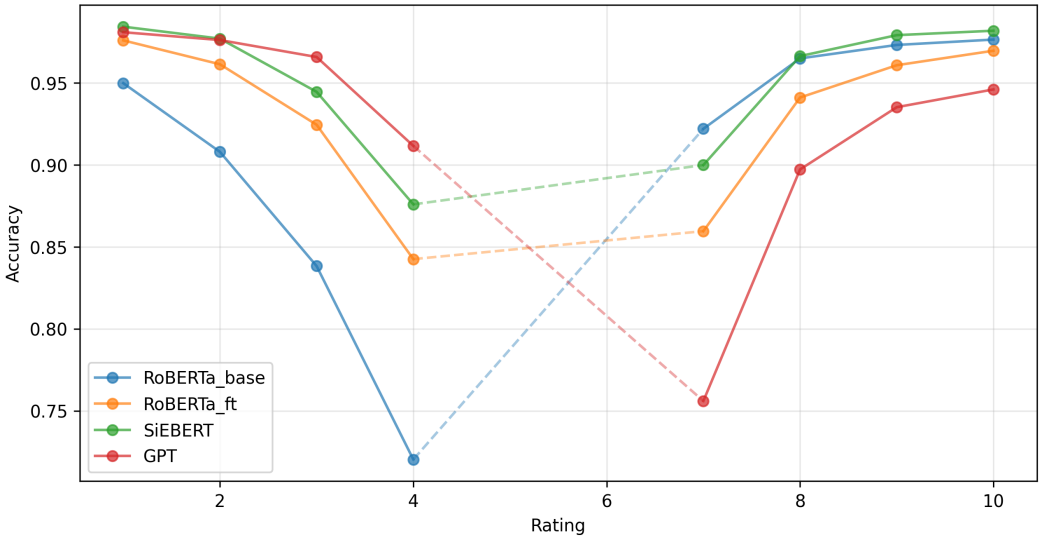

Figure 2: Accuracy by rating

5.5.1 By ratings

A clear drop in performance is evident for ratings closer to neutral, as expected (see Figure 2). However, the models exhibit notably different behaviors. The accuracy of RoBERTa-base and GPT-3.5 Turbo varies with the valence of reviews: RoBERTa-base performs significantly better on positive reviews than on negative ones, whereas GPT-3.5 Turbo shows the opposite trend, excelling on negative reviews. These models also experience the largest performance declines on their weaker side: -31.9% for RoBERTa-base and -25.1% for GPT-3.5 Turbo. Fine-tuned RoBERTa models, on the other hand, maintains similar performance for both negative and positive reviews, with comparable declines near neutral ratings. This consistency makes them not only the best-performing models but also the most stable across sentiment extremes [6] .

5.5.2 By review length

Slightly different observations emerge: SiEBERT maintains stable performance, while other models show a clear downward trend, though the effect of review length is much smaller than that of review valence (see Figure 3). This aligns with the observation that truncation had minimal impact, suggesting that the key information in reviews is not heavily dependent on their length.

6 Discussion

The question addressed in this report was: is the IMDb dataset a limiting factor, responsible for the plateauing of SOTA performance?

[6]Regarding the GPT model, providing more positive examples than negative one in the prompt could have helped balance its performance.

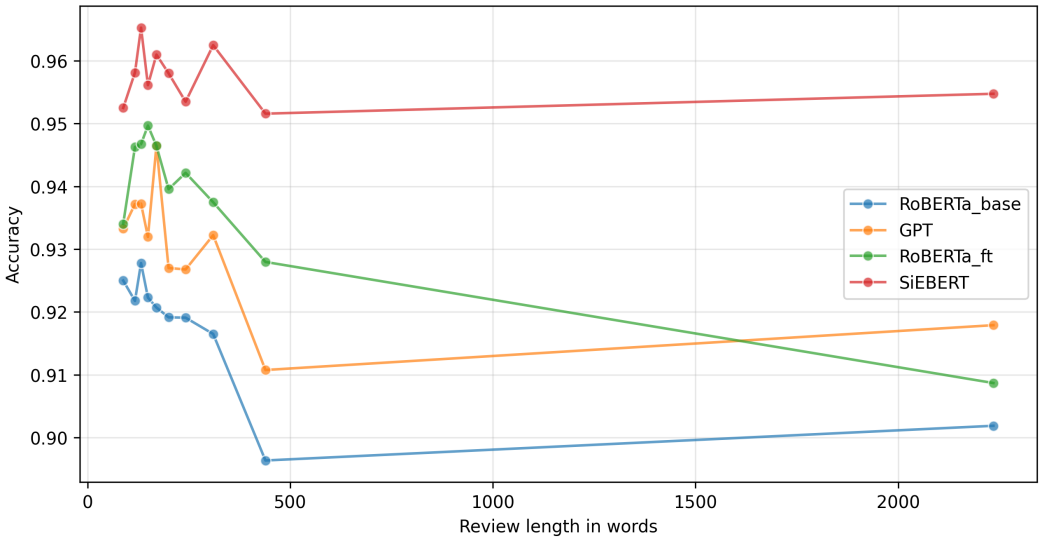

Figure 3: Accuracy by review length

The first hypothesis—that noise in the true labels arises because ratings are an imperfect proxy for sentiment—is supported by the fact that accuracy drops significantly for all models when classifying reviews associated with ratings of 4 and 7. One could argue that this phenomenon is perfectly normal: these reviews are harder to classify because they express more nuanced sentiments and are less polarized, making them intrinsically more challenging for models. However, the dataset's authors somewhat undermine this argument by explicitly excluding reviews deemed too close to neutral, with the explicit aim of focusing on highly polarized reviews. So maybe reviews associated with ratings of 4 and 7 should be excluded as well. This illustrates a fundamental challenge in constructing labeled datasets. Automating the labeling process is tempting, as manual annotation can be as tedious as building a sentiment lexicon. However, automation carries the risk of producing inaccurate or inadequate labels. In such circumstances, an insightful extension would have been to include reviews spanning the full range of ratings and then: (1) examine whether models indeed struggle more with reviews associated with ratings of 5 and 6, and (2) introduce a third category for ternary classification. This category could be defined as "mixed" instead of "neutral", which could prove especially useful for reviews expressing conflicted sentiments ("I really liked the plot, but the acting was catastrophic…").

The second hypothesis—that longer reviews are harder to classify—is also confirmed, albeit the corresponding effect is much smaller. The logical extension here would be to shift toward aspect-level sentiment analysis, identifying which aspects of a movie are viewed positively and which are viewed negatively. While significantly more complex, this would be more insightful than simply characterizing the overall tone, and likely represents the future of sentiment analysis for lengthy documents.

References

[1] Myriam Munezero, Calkin Suero Montero, Erkki Sutinen, and John Pajunen. Are They Different? Affect, Feeling, Emotion, Sentiment, and Opinion Detection in Text. *IEEE Transactions on Affective Computing*, 5(2):101–111, April 2014. ISSN 1949-3045. doi: 10.1109/TAFFC.2014.2317187. URL `https://ieeexplore.ieee.org/document/6797872`. Conference Name: IEEE Transactions on Affective Computing.

[2] Robert A. Stine. Sentiment Analysis. *Annual Review of Statistics and Its Application*, 6:287–308, March 2019. ISSN 2326-8298, 2326-831X. doi: 10.1146/annurev-statistics-030718-105242. URL `https://www.annualreviews.org/content/journals/10.1146/annurev-statistics-030718-105242`. Publisher: Annual Reviews.

[3] Kumar Ravi and Vadlamani Ravi. A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. *Knowledge-Based Systems*, 89:14–46, November 2015. ISSN 0950-7051. doi: 10.1016/j.knosys.2015.06.015. URL https://www.sciencedirect.com/science/article/pii/S0950705115002336.

[4] Fatemeh Hemmatian and Mohammad Karim Sohrabi. A survey on classification techniques for opinion mining and sentiment analysis. *Artificial Intelligence Review*, 52(3):1495–1545, October 2019. ISSN 1573-7462. doi: 10.1007/s10462-017-9599-6. URL https://doi.org/10.1007/s10462-017-9599-6.

[5] Sudhanshu Kumar, Partha Pratim Roy, Debi Prosad Dogra, and Byung-Gyu Kim. A Comprehensive Review on Sentiment Analysis: Tasks, Approaches and Applications, November 2023. URL http://arxiv.org/abs/2311.11250. arXiv:2311.11250 [cs].

[6] Jochen Hartmann, Mark Heitmann, Christian Siebert, and Christina Schamp. More than a Feeling: Accuracy and Application of Sentiment Analysis. *International Journal of Research in Marketing*, 40(1):75–87, March 2023. ISSN 0167-8116. doi: 10.1016/j.ijresmar.2022.05.005. URL https://www.sciencedirect.com/science/article/pii/S0167811622000477.

[7] C. Hutto and Eric Gilbert. 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, May 2014. ISSN 2334-0770. doi: 10.1609/icwsm.v8i1.14550. URL https://ojs.aaai.org/index.php/ICWSM/article/view/14550. Number: 1.

[8] Zengzhi Wang, Qiming Xie, Yi Feng, Zixiang Ding, Zinong Yang, and Rui Xia. Is ChatGPT a Good Sentiment Analyzer? A Preliminary Study, February 2024. URL http://arxiv.org/abs/2304.04339. arXiv:2304.04339 [cs].

[9] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training, June 2018. URL https://openai.com/index/language-unsupervised/.

[10] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners, February 2019. URL https://openai.com/index/better-language-models/.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. URL http://arxiv.org/abs/1810.04805. arXiv:1810.04805 [cs].

[12] Pawanjit Singh Ghatora, Seyed Ebrahim Hosseini, Shahbaz Pervez, Muhammad Javed Iqbal, and Nabil Shaukat. Sentiment Analysis of Product Reviews Using Machine Learning and Pre-Trained LLM. *Big Data and Cognitive Computing*, 8(12):199, December 2024. ISSN 2504-2289. doi: 10.3390/bdcc8120199. URL https://www.mdpi.com/2504-2289/8/12/199. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[13] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1015/.

[14] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach, July 2019. URL http://arxiv.org/abs/1907.11692. arXiv:1907.11692 [cs].

[15] Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. Smarter, Better, Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Context Finetuning and Inference, December 2024. URL http://arxiv.org/abs/2412.13663. arXiv:2412.13663 [cs].

[16] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.