Spoiler Alert: Using Deep Learning to Detect Spoilers in Text and Generate Similar Spoiler-Free Text

Abolfazl Eshagh, Mobina Salimipanah, Parnian Razavipour, Ali Nikkhah, Sarina Zahedi, Ramtin Khoshnevis

Abstract

This project addresses the challenge of managing spoilers in movie reviews through two tasks: spoiler-free text generation and spoiler classification. The increasing prevalence of spoilers in online content makes it critical to develop tools that can safeguard the narrative experience for users. Using the IMDB Spoiler Dataset, which contains both spoiler-free plot summaries and full plot synopsis with spoilers, we aim to contribute to this goal by refining models capable of generating spoiler-free movie plots. We explored and trained several advanced transformer-based models, including BART, LED, BigBird, and LongT5, to generate concise, coherent movie summaries without revealing key plot twists. For the classification task, we developed models to accurately distinguish spoiler-containing reviews from spoiler-free ones, combining traditional NLP techniques and deep learning methods. The models' effectiveness was assessed through standard NLP evaluation metrics, demonstrating promising results in both tasks. Our project not only advances spoiler detection but also enhances user satisfaction by providing spoiler-free summaries, offering a balanced solution to preserve narrative suspense.

1. Introduction

A spoiler is any piece of information that reveals key plot points or critical details about a narrative, often ruining the suspense or surprise for someone who hasn't yet experienced the work. Spoilers can come in many forms, from social media posts to reviews or even casual conversations, and they can significantly diminish the enjoyment of a book, movie, or TV show. As media consumption has become more widespread and instantaneous, the challenge of avoiding spoilers has grown, making it necessary to develop tools that can detect and mitigate them.

Natural Language Processing (NLP) offers a powerful approach to tackling the spoiler detection problem because it allows machines to understand, analyze, and manipulate human language in a way that can discern which parts of a text may reveal critical information. By training deep learning models on labeled data containing spoilers and non-spoilers, we can teach the model to identify patterns and contexts that often indicate spoilage. Moreover, NLP can go beyond detection, potentially enabling the generation of spoiler-free alternatives by rewriting or summarizing text in a way that preserves the essence of the content without revealing critical plot points.

Transformer models, such as BERT and GPT, are particularly effective in handling spoiler detection and text generation tasks because of their ability to understand context and maintain long-range dependencies within the text. These models, pre-trained on vast datasets, are capable of detecting subtle nuances in language that indicate spoilers. Using this technology, we built a robust platform that not only detects spoilers in text but also generates spoiler-free versions of the same content. The platform ensures that users can engage with reviews, summaries, or discussions without having the core narrative spoiled, providing a seamless and enjoyable experience.

https://github.com/NLP-Final-Projects/Spoiler-Alert

This project addresses the dual tasks of spoiler detection and spoiler-free content generation, using deep learning models to achieve these goals. We will provide more details on our methodology in later sections, but in this introduction, we focus on the related work that has been done in this field and how our approach compares.

Several studies have explored the problem of spoiler detection, often focusing on different media types or datasets.

Bao et al. (2021) [1] used the **UCSD Goodreads Spoiler dataset**, which consists of user reviews of books from the Goodreads website. In their study, they focused on sentence-level spoiler detection, applying models like **LSTM, BERT, and RoBERTa**. Their approach relied on deep learning models to automatically detect spoilers without the need for handcrafted features, contrasting with an earlier UCSD team's work that used handcrafted features such as Document Frequency-Inverse Item Frequency (DF-IIF) for spoiler detection. Bao et al. were able to achieve results comparable to the original UCSD team by training on sentence-level data extracted from reviews, with about 270,000 samples in the training set.

Zeng et al. (2023) [2] introduced the **MMoE (Multi-modal Mixture-of-Experts)** framework, designed to handle genre-specific spoilers in movie reviews. They used a combination of **textual, metadata, and graph features** from movie reviews, user interactions, and movie metadata to enhance spoiler detection. Their model integrated data from multiple modalities, making it more robust in detecting spoilers that depend heavily on context, such as movie genres and user preferences. MMoE achieved state-of-the-art results on the **Kaggle IMDB Spoiler dataset** and the **LCS dataset**, surpassing previous approaches like **SpoilerNet** and **MVSD** in terms of accuracy and F1 score(2403.05265v2-2). While SpoilerNet relied solely on text-based methods, Zeng et al. emphasized the importance of metadata and user information, making their approach more comprehensive.

Heng et al. (2023) [3] extended this work with their **MVSD (Multi-View Spoiler Detection)** framework, which incorporated external movie knowledge and user interactions to detect spoilers more effectively. They used **Graph Neural Networks (GNNs)** and focused on leveraging user profiles and movie-specific data to enhance spoiler detection accuracy. Like MMoE, MVSD demonstrated the need to move beyond simple text-based methods, showing that the integration of multi-modal information leads to better results.

In contrast to these works, our project tackles both spoiler detection and spoiler-free text generation, focusing on **movie reviews and plots** and using the **IMDB Spoiler dataset**. In addition to training transformer-based models for spoiler-free generation, we extracted a variety of features in our classification task to improve spoiler detection. These features, which are discussed in more detail in the method section, include specific characteristics such as the presence of spoiler-related keywords and sentence length. These features were integrated into our model training to enhance detection accuracy.

While previous studies like MMoE and MVSD focus heavily on spoiler detection using multi-modal data, our approach goes a step further by also addressing the challenge of generating coherent, concise summaries of movie plots without revealing critical spoilers. We fine-tuned several transformer-based models—**BART, LED, BigBird, and LongT5**—to achieve high performance in this task. Our project's novelty lies in this dual approach, as well as in the diversity of the models we employ.

2. Datasets

For this work, we utilized the IMDB Spoiler Dataset, which is divided into two parts:

1. **IMDB Reviews Dataset**: This dataset contains 573,913 user reviews from the IMDb website, where users have the ability to annotate their reviews as containing spoilers or not. Out of the total reviews, 150,924 (~26.3%) are marked as spoilers. We used this dataset for both review summary generation and spoiler detection classification tasks. Specifically, for these

tasks, we utilized the review_text and is_spoiler columns to identify and classify spoiler-related content.

Each record in IMDB_reviews consists of the following attributes:

- review_date: date the review was posted.
- movie_id: unique id of the movie for which the review is about.
- user_id: unique id of the user who left the review.
- is_spoiler: whether the review contains spoilers.
- review_text: text of the review.
- rating: star rating (outof10) given by the user.
- review_summary: review summary provided by the user.

2. **Movie Plot Synopsis Dataset**: This dataset is part of the IMDB Spoiler Dataset and focuses on movie metadata, including plot summaries and synopses. The dataset contains two key columns: plot_summary, which provides a spoiler-free summary of the movie, and plot_synopsis, which contains detailed plot descriptions that include spoilers. Rows without values in the plot_synopsis column were removed during preprocessing. The plot_summary column was utilized for spoiler-free summary generation, while the plot_synopsis was employed in generating detailed summaries and spoiler detection.

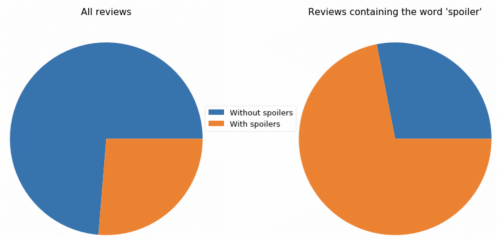
Each record in IMDB_movie_details consists of the following attributes:

- movie_id: unique id for the movie.
- plot_summary: summary of the plot without any spoilers.
- duration: runtime of movie.
- genre: list of genres the movie belongs to.
- rating: star rating out of 10.
- release_date: date the movie was released.
- plot_synopsis: revealing details of the plot of the movie.

Statistic	Value
# reviews	573,913
# spoiler reviews	150,924
# users	263,407
# movies	1,572
# users with at least one spoiler review	79,039
# movies with at least one spoiler review	1,570

Table 1: General statistics of the dataset.

In order to visualize how reliable the spoiler annotation on IMDb is, we simply visualize the spoiler/non-spoiler annotated reviews that contain the word “spoiler”. [2]



Also in the following table we can see the average length (number of tokens), max length and count in each used columns:

statistics	Average length	Max length	Count
Review text	1460	14963	573913
Plot summary	151	315	1572
Plot synopsis	2117	18103	1572

Table 2: Data Exploration

3. Method

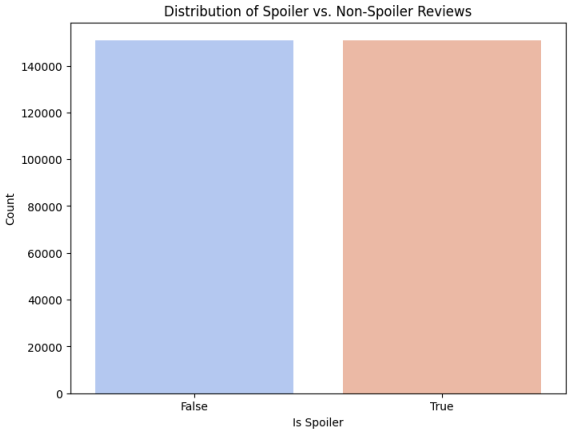
Dataset preprocessing

The first step involved cleaning and preparing the datasets for model training. For both datasets, we

removed irrelevant columns and filtered out any entries with missing values in key columns such as plot_synopsis and review_summary.

Also we implemented a text preprocessing function to normalize and standardize the data by converting all text to lowercase, removing special characters, and replacing newlines and extra spaces. This was applied to the `review_text`, `Plot Synopsis' and `Plot Summary` columns to prepare the data for analysis.

We used down-sampling methods to make two equally distributed classes for better classification training.



4.1. Classification

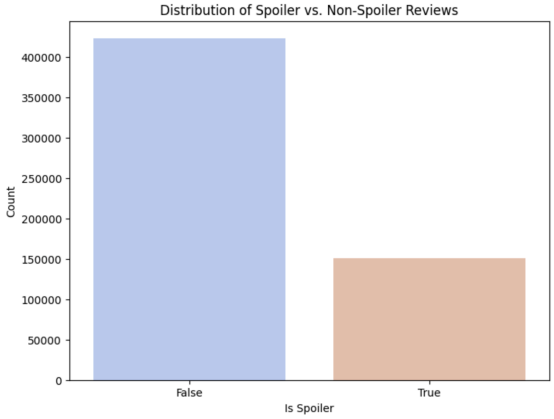
In the classification task, We have 2 main parts. First Preprocessing and Feature Extraction. Then Training different models and using different methods like voting between models

Feature Extraction

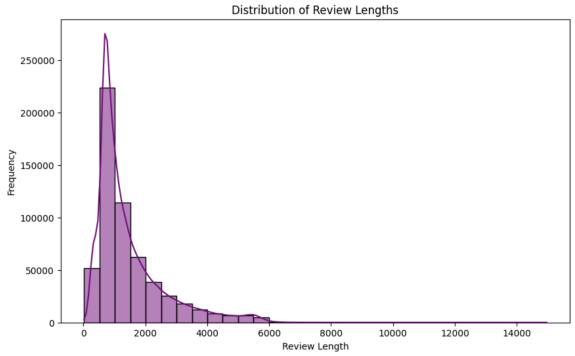
Features are crucial in machine learning. Here, they are extracting specific characteristics from the reviews that will help in classification.

To achieve this, we utilized a dataset obtained from Kaggle and extracted a range of features for analysis. Our initial feature creation involved adding a column titled 'bad_flag', which indicates the presence of certain stemmed spoiler-related words (e.g., "die", "kill") within the tokenized and stemmed version of each review.

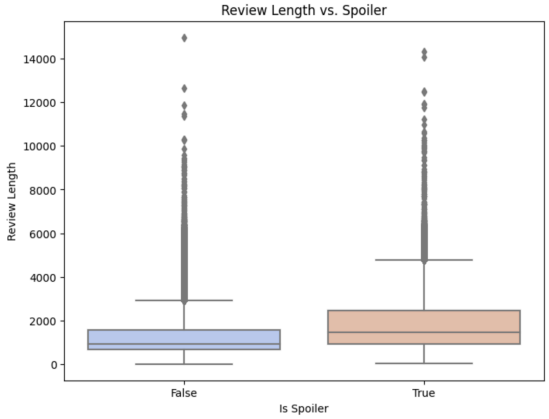
Considering the plots below, we got inspired to generate more features and columns.



The length distribution shows us we can barely consider length as an effective feature.



But here, It shows maybe longer reviews willing to be spoiler more and we can consider it.



Additionally, we developed two alternative scoring methods to evaluate each review independently of the specific movie being discussed. This scoring system

involves assigning a weighted score to the reviews, followed by normalization to standardize the values. Through these approaches, we generated three distinct columns of data. These features were subsequently used to train our models, which we will elaborate on in the following sections of this paper, allowing us to monitor improvements in model accuracy over time.

Then we use these preprocessed datasets to improve training.

Training Models

Our training procedure began with a straightforward LSTM model, which served as our baseline for spoiler detection. The LSTM was chosen for its ability to handle sequential data, capturing temporal dependencies within the text. While it delivered reasonable performance, we recognized the limitations of LSTMs, particularly in handling long-range dependencies and context within the data. To address these limitations, we moved on to more sophisticated Transformer-based models like BERT, RoBERTa, and T5. These models are designed to understand context better, thanks to their attention mechanisms, which allow them to consider the importance of different words in relation to one another across the entire text. As a result, these models outperformed the LSTM, with RoBERTa achieving the highest accuracy on our validation set.

One of the key advantages of Transformer models over LSTM is their ability to capture complex language patterns and relationships within large text sequences. BERT and RoBERTa, in particular, excel at understanding the context of each word within a sentence, making them more effective at tasks like spoiler detection. T5, on the other hand, frames every problem as a text-to-text task, providing flexibility in fine-tuning for various NLP tasks. However, these models are significantly more computationally intensive, requiring substantial GPU memory. Due to the limited GPU memory available, we had to implement techniques like gradient accumulation, which allowed us to effectively simulate larger batch sizes by accumulating gradients over several mini-batches before updating the model weights. Additionally, we had to reduce the batch size to fit within the GPU memory constraints, which slowed down training but was necessary to avoid memory overflow.

Another challenge we encountered was the size of the training text. With an average length of just over 4,000 tokens, the reviews were too long to be processed directly by models like BERT, which has a maximum token limit of 512. To work around this, we had to chunk the text into smaller segments, each within the token limit, and train the model on these chunks. This approach allowed us to leverage BERT's capabilities, but it also required careful handling to ensure that the context was preserved across chunks. Despite these challenges, the Transformer models, particularly RoBERTa, demonstrated significant improvements over the LSTM baseline, validating our decision to adopt more advanced architectures for this task.

We trained our best models, including the fine-tuned RoBERTa, on a single P100 GPU available on Kaggle for approximately 8 hours. Throughout this training process, we observed a consistent decrease in loss across all epochs without signs of overfitting. This steady improvement suggests that our models were effectively learning from the data. Given these results, it is likely that further training on a more powerful setup could yield even better performance in the future, potentially pushing our validation accuracy beyond its current level.

4.2. Generation

In the generation phase, we have undertaken multiple tasks, focusing on generating spoiler-free plot summaries using four different models for long text generation tasks: BART, LED, BigBird and LongT5… we fine tuned these models using T4 GPU on the colab and P100 on the kaggle.

Spoiler-Free Plot Summary Generation using BART

The BART model (Bidirectional and Auto-Regressive Transformers) was employed to generate spoiler-free plot summaries. This model is particularly effective for text generation tasks due to its encoder-decoder architecture, which allows it to understand and generate coherent text sequences.

For this task, we performed initial preprocessing on our dataset. It is important to note that some entries lacked values in the plot_synopsis column, necessitating the removal of those samples. This ensured that the dataset used for training was clean and complete.

We defined our model using the BartForConditionalGeneration model, specifically the facebook/bart-large configuration. The BartTokenizer was utilized to preprocess the data. Each synopsis was tokenized into sequences of up to 512 tokens, while the summaries were tokenized to a maximum of 128 tokens. This tokenization ensures that the inputs and outputs are formatted correctly to feed into the BART model while preserving the core narrative structure.

We fine-tuned the BartForConditionalGeneration model over 10 epochs. The initial learning rate was set to 1.0000000000000002e-06. Given the limited size of our available dataset, the model quickly overfitted, making further fine-tuning beyond 10 epochs impractical. The best state of the model, captured during this training session, was saved and utilized for evaluation.

The training setup involved framing the input synopsis as a summarization task, with each input prefixed by "summarize: " to prompt the model for summary generation. The AdamW optimizer was employed to update the model's parameters, balancing computational efficiency with convergence. The fine-tuning process was designed to adapt the model to generate spoiler-free summaries from synopses, leveraging BART's ability to understand and compress text into shorter, more digestible summaries.

The training arguments were configured with a batch size of 2 per device, a weight decay of 0.01, and a warmup of 500 steps to prevent overfitting. The evaluation strategy was set to evaluate the model every 500 steps, saving the best model based on evaluation performance.

Despite the challenges posed by the small dataset, the BART model demonstrated its capacity to generate coherent and concise plot summaries. The training process highlighted the model's strengths in text generation, although the issue of overfitting due to the limited data size was a significant constraint.

Spoiler-Free Plot Summary Generation with LED

The LED model (Longformer Encoder-Decoder) was introduced as an alternative to handle longer texts, making it ideal for processing extensive movie synopsis, which often contain detailed and spoiler-rich information. The LED model's architecture is based on the Longformer, which is designed to handle extremely long sequences by using sparse attention, making it highly suitable for tasks requiring comprehension of lengthy documents.

For this task, we utilized the LEDTokenizer to preprocess the data. Each synopsis was tokenized into sequences of up to 3000 tokens, which represents a large portion of text that the LED model can handle in a single forward pass. The summaries were tokenized to a maximum of 1024 tokens, reflecting the goal of producing concise and spoiler-free plot summaries from these longer inputs. This tokenization ensures that the inputs and outputs are formatted correctly to feed into the LED model while preserving the core narrative structure.

We fine-tuned the LEDForConditionalGeneration model, pre-trained on the allenai/led-base-16384 configuration. This pre-trained model, which was initially trained on large text documents, provides a strong foundation for summarizing lengthy texts. During fine-tuning, the LED model was trained on the dataset for 3 epochs, using a learning rate of 2e-5 and a weight decay of 0.01 to prevent overfitting. Given the large size of the model, we utilized a small batch size of 1 per device due to its high memory requirements.

The training setup involved framing the input synopsis as a summarization task, with each input prefixed by "summarize: " to prompt the model for summary generation. The AdamW optimizer was employed to update the model's parameters, balancing computational efficiency with convergence. The fine-tuning process was designed

to adapt the model to generate spoiler-free summaries from synopses, leveraging the LED's ability to understand and compress large bodies of text into shorter, more digestible summaries.

This setup demonstrated the LED model's capacity to efficiently generate non-spoiler plot summaries from longer movie synopsis. Despite the challenges of working with such large sequences, the LED model managed to effectively synthesize key details from the input while avoiding the inclusion of spoiler content. The training process, however, was limited to 3 epochs due to the high computational demands of the model

Spoiler-Free Plot Summary Generation with BigBird

The BigBird model, developed by Google Research, is an adaptation of the Transformer architecture that is specifically optimized for handling long sequences of text. Unlike traditional Transformers that struggle with long documents due to their quadratic computational cost related to sequence length, BigBird utilizes a sparse attention mechanism. This innovation allows it to efficiently process sequences much longer than standard models can handle, making it particularly suitable for applications like document summarization where context from a large body of text is crucial.

For the task of generating non-spoiler summaries from movie plot synopsis, we employed the BigBird model because of its ability to manage the extensive length of such texts effectively. Using the BigBirdTokenizer, we first converted the texts from our dataset, which included fields for both detailed plot synopsis and concise summaries, into a form suitable for the model. This involved truncating and padding the sequences to fixed lengths — 3000 tokens for synopses and 315 for summaries — to maintain uniformity in input size.

The model of choice, BigBirdPegasusForConditionalGeneration, was pre-trained on the 'arxiv' subset of documents, well-suited for academic-style writing and lengthy content. To adapt this model to our summarization task, we fine-tuned it on our dataset using the TrainingArguments configuration. The fine-tuning was conducted over 3 epochs with a batch size of 4 and a learning rate of 5e-05, complemented by a weight decay of 0.01, to effectively balance learning accuracy without overfitting. Employing the Trainer API, the model was trained using plot synopsis as inputs and the corresponding non-spoiler summaries as targets, which enabled it to learn the production of concise and relevant summaries that avoid spoilers.

This approach leveraged BigBird's robust handling of long texts to produce precise and relevant summaries, thus demonstrating the model's capability in a practical application beyond its typical academic use cases. The model's performance was periodically evaluated on a validation set to ensure that the training process was effective and that the model generalized well to unseen data.

By fine-tuning BigBird on this specialized task, we demonstrated its utility in a real-world application, underscoring the model's adaptability and effectiveness in dealing with extensive textual data within the domain of entertainment content summarization.

Spoiler-Free Plot Summary Generation with Long T5

We employed the LongT5 model, optimized for longer text sequences, to generate non-spoiler summaries from detailed movie plots. Using the google/long-t5-local-base variant and its corresponding AutoTokenizer, we prepared the text by truncating and padding the plot synopses to 3500 tokens and summaries to 315 tokens. This preparation ensured uniform input sizes for model training. We fine-tuned LongT5 on our dataset with specific training parameters: 2 epochs, training and evaluation batch sizes of 2 and 4 respectively, a learning rate of 2e-05, and weight decay of 0.01. These settings, along with gradient accumulation and warmup steps, were calibrated to optimize learning without overfitting. The entire training process was managed using the Hugging Face Trainer API, which facilitated efficient model iterations and evaluations. This approach demonstrated LongT5’s capability to

handle extensive textual data, producing accurate and relevant summaries without spoilers.

Review Summary Generation

The BART model (Bidirectional and Auto-Regressive Transformers) was employed to generate review summaries. This model is particularly effective for text generation tasks due to its encoder-decoder architecture, which allows it to understand and generate coherent text sequences.

Similar to the plot summary generation process, we began by preprocessing this part of the dataset. The only two features needed were the review_text and review_summary columns, so all other columns were discarded. It is worth mentioning that this part of the dataset is significantly larger, containing more than 570,000 samples of review text and corresponding summaries. Due to computational resource constraints, we had to downsample this dataset randomly.

We defined our model using the BartForConditionalGeneration model, specifically the facebook/bart-large configuration. The BartTokenizer was utilized to preprocess the data. Each review text was tokenized into sequences of up to 512 tokens, while the summaries were tokenized to a maximum of 128 tokens. This tokenization ensures that the inputs and outputs are formatted correctly to feed into the BART model while preserving the core narrative structure.

We fine-tuned the BartForConditionalGeneration model over 10 epochs. The initial learning rate was set to 1.0000000000000002e-06. Unlike the plot summary task, the larger dataset size allowed the BART model to continue achieving lower losses with each epoch, indicating better generalization and reduced overfitting.

The training setup involved framing the input review text as a summarization task, with each input prefixed by "summarize: " to prompt the model for summary generation. The AdamW optimizer was employed to update the model's parameters, balancing computational efficiency with convergence. The fine-tuning process was designed to adapt the model to generate concise and coherent review summaries, leveraging BART's ability to understand and compress text into shorter, more digestible summaries.

The training arguments were configured with a batch size of 2 per device, a weight decay of 0.01, and a warmup of 500 steps to prevent overfitting. The evaluation strategy was set to evaluate the model every 500 steps, saving the best model based on evaluation performance.

The larger dataset size for the review summary task allowed the BART model to demonstrate its capacity to generate coherent and concise summaries effectively. The training process highlighted the model's strengths in text generation, with continuous improvement in performance across epochs.

4. Metrics and Results

Classification:

The baseline model employed in our project is a bidirectional LSTM neural network (model available due to request) built using TensorFlow's Keras API. The model starts with an embedding layer that maps input words into dense vectors of fixed size, followed by two bidirectional LSTM layers, each with 64 units. This architecture is designed to capture both past and future context in the input sequences, making it well-suited for tasks requiring sequential data processing. To prevent overfitting, dropout layers are added after each LSTM layer. The final dense layer with a sigmoid activation outputs a probability score for binary classification. Over eight epochs, the model demonstrated a progressive increase in accuracy, reaching *80.45% on the training set*, with a validation accuracy plateauing around *76%*. Despite slight fluctuations in validation loss, the model provided a solid starting point for further experimentation.

In addition to the LSTM model, we fine-tuned three transformer-based models—BERT, T5, and RoBERTa—for the spoiler detection task. Each model was pre trained on large-scale language data, making them highly effective for natural language understanding. For BERT, we employed a base uncased model and adjusted its hyperparameters to

optimize performance on our dataset. T5, a text-to-text model, was fine-tuned to generate labels based on input reviews. RoBERTa, a robustly optimized version of BERT, was also fine-tuned for binary classification. Among these models, RoBERTa delivered the highest accuracy on the validation set, achieving *an impressive 87%*. This result underscores RoBERTa's capability in capturing nuanced language patterns, making it the most effective model for our spoiler detection classifier.

In addition to training machine learning models, we implemented rule-based methods to detect spoilers by flagging reviews that contained specific predefined words often associated with key plot revelations, such as *"murder," "death," "betrayal," and "secret.*" These words were chosen based on their frequent association with critical plot points in movies. To further enhance our detection system, we experimented with different ensemble techniques, including majority voting among our LSTM, BERT, T5, and RoBERTa models, alongside the rule-based method. However, these ensemble approaches did not yield significant improvements in overall accuracy or spoiler detection performance, indicating that our model-based approach, particularly with RoBERTa, was already capturing the necessary context effectively.

	BERT	RoBERTa	T5	LSTM
accuracy	82 %	**87%**	78%	75%

Table 1. The results for *review classification.*

Spoiler-Free Summary Generation:

The performance of our models was rigorously evaluated using standard metrics in natural language processing, specifically ROUGE and BLEU scores. ROUGE scores assess the overlap between the generated summaries and reference summaries, measuring precision, recall, and F-measure. BLEU scores focus on the precision of n-grams in the generated text compared to the reference, offering insights into the fluency and coherence of the summaries.

Table 3 presents a comparison of the ROUGE and BLEU scores for the different models: BART, LED, BigBird, and LongT5. The LED model demonstrated notable strengths, particularly in terms of the BLEU score, which indicates its superiority in producing coherent and fluent summaries. This can be attributed to LED's ability to handle longer contexts more effectively than the other models. The LED model achieved the highest BLEU score of 0.0348, significantly outperforming BART, which scored 0.0163 in this metric.

In contrast, BART slightly outperformed LED in terms of ROUGE scores, particularly ROUGE-1 and ROUGE-L. These metrics suggest that BART generated summaries with a higher degree of overlap with the reference summaries, which is likely a reflection of its fine-tuning on smaller and medium-sized datasets. However, BART's performance in generating longer summaries was slightly less consistent when compared to LED.

BigBird and LongT5, while capable models, did not perform as well as BART and LED. BigBird achieved moderate ROUGE scores but lagged behind in BLEU. Similarly, LongT5 demonstrated the lowest scores across all metrics, indicating that it struggled to produce summaries with both high overlap and fluency compared to the other models.

This comparison highlights that LED, with its capability to process long sequences effectively, is well-suited for summarization tasks involving extensive textual content. On the other hand, BART's performance in ROUGE metrics makes it a strong contender for tasks that prioritize accuracy in shorter summaries. In summary, the choice between these models depends on the specific requirements of the task—whether longer context comprehension or precise overlap is more critical.

In our study, we implemented a Long Short-Term Memory (LSTM) neural network to classify text reviews. Our model architecture was tailored to capture the temporal dependencies inherent in textual data, making it well-suited for complex classification tasks such as distinguishing between spoiler and non-spoiler content. We fine-tuned this LSTM model on a substantial dataset comprising 175,000 text

reviews, optimizing our network to accurately understand and classify textual nuances related to content spoilers.

Following the fine-tuning process, we employed the trained model to evaluate summaries generated by our summary generation system. This validation step was critical to ensure that the summaries provided to end-users were free of spoilers. The results were promising: our classification model correctly identified the generated summaries as non-spoilers, confirming the efficacy of both our summary generation and classification methodologies. This dual-step validation not only enhanced the reliability of our summarization output but also demonstrated the LSTM model's robustness in handling real-world text classification tasks.

Review Summary Generation:

The performance of our BART model for review summary generation was evaluated using the same metrics—ROUGE and BLEU scores—that were applied to spoiler-free summary generation. As shown in Table 2, the BART model achieved a ROUGE-1 score of 0.1110, a ROUGE-2 score of 0.0415, and a ROUGE-L score of 0.1067. The BLEU score for this task was 0.0219.

These results indicate that while the BART model is capable of generating summaries with a reasonable degree of overlap with the reference summaries (as reflected in the ROUGE scores), its performance in terms of fluency and n-gram precision, as captured by the BLEU score, is modest. This suggests that the model might still struggle with capturing the finer details necessary for generating highly fluent and coherent summaries after only one epoch of training.

	ROUGE1	ROUGE2	ROUGEL	Blue
BART	0.1110	0.0415	0.1067	0.0219

Table 2. The results for *review summary* generation using *BartForGeneration* on multiple metrics. (after 1 epoch)

	ROUGE1	ROUGE2	ROUGEL	Blue
BART	**0.2744**	**0.0555**	**0.1685**	0.0163
LED	0.2699	0.0550	0.1656	**0.0348**
BigBird	0.1844	0.0122	0.1306	-
LongT5	0.1513	0.0078	0.1073	-

Table 3. presents a detailed comparison of the BART, LED, BigBirtd and longT5 models, emphasizing their respective ROUGE and BLEU scores.

5. Conclusion

This study has successfully tackled two main objectives: spoiler detection and the generation of spoiler-free summaries, using advanced deep learning models. We discovered that transformer-based models like RoBERTa significantly outperform traditional LSTMs in identifying spoilers, achieving up to 87% accuracy. RoBERTa's nuanced language comprehension demonstrates the effectiveness of attention mechanisms in processing complex textual data.

In generating spoiler-free content, the LED model excelled, particularly with extensive texts, as evidenced by its high BLEU scores. This capability is essential for producing coherent summaries of lengthy narratives without revealing plot details. Although BART also performed well, particularly in ROUGE metrics, the LED model's adaptability to long texts makes it especially valuable for comprehensive summarizations.

Our findings not only advance the academic understanding of text processing but also offer practical solutions for media platforms to enhance user interactions without spoilers. Looking ahead, refining these models and exploring hybrid approaches could further improve their efficacy and extend their applicability to various languages and contexts, broadening their impact on global media consumption.

6. Future works

There are several directions for improving the current project. First, training the models with a larger dataset and more powerful computational resources could significantly enhance performance, especially in generating more nuanced and accurate spoiler-free summaries. Additionally, exploring more complex models and analyzing their performance under different conditions could further improve the accuracy and generalization of both the spoiler detection and generation tasks.

7. References

[1] Allen Bao, Marshall Ho, and Saarthak Sangamnerkar "Using Natural Language Processing to Detect Spoilers in Book Reviews" , 2021. [Online]. Available: https://arxiv.org/pdf/2102.03882

[2] Heng Wang, Wenqian Zhang and Yuyang Bai " Detecting Spoilers in Movie Reviews with External Movie Knowledge and User Networks"

[3] Zinan Zeng, Sen Ye, Zijian Cai and Heng Wang " **MMoE**: Robust Spoiler Detection with Multi-modal Information and Domain-aware Mixture-of-Experts"

[4] R. Misra, "IMDB Spoiler Dataset," Kaggle, 2020. [Online]. Available: https://www.kaggle.com/datasets/rmisra/imdb-spoiler-dataset. [Accessed: Aug. 10, 2024].

[5] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," in *Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 7871-7880. [Online]. Available: https://arxiv.org/abs/1910.13461

[6] "Bart — transformers 4.25.1 documentation," Hugging Face. [Online]. Available: https://huggingface.co/docs/transformers/en/model_doc/bart . [Accessed: Aug. 10, 2024].

[7] "IMDB Spoiler Flagged," Kaggle, 2024. [Online]. Available:https://www.kaggle.com/datasets/alinikkhah2001/bad-words-flag [Accessed: Aug. 16, 2024].

[8] "IMDB Spoiler Scored1," Kaggle, 2024. [Online]. Available:https://www.kaggle.com/datasets/alinikkhah2001/final_dataset20 [Accessed: Aug. 16, 2024].

[9] "IMDB Spoiler Scored2," Kaggle, 2024. [Online]. Available:https://www.kaggle.com/datasets/alinikkhah2001/final_dataset21 [Accessed: Aug. 16, 2024].

[10] Wan, M., Misra, R., Nakashole, N., & McAuley, J., “Fine-Grained Spoiler Detection from Large-Scale Review Corpora”, 2019, In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (pp. 6534-6543). [Online]. Available: https://aclanthology.org/P19-1248.pdf