Amazon Reviews Sentiment Analysis Report

Deep Learning Project

May 2024

Abstract

This report investigates the application of binary sentiment analysis on Amazon review texts. Using a dataset comprising 3.6 million reviews, we employed various neural network architectures, including Convolutional Neural Networks (CNN), Long Short-Term Memory networks (LSTM), a hybrid LSTM-Gated Recurrent Unit (GRU) model, and a transformer-based Distil BERT model. Our aim was to evaluate and compare the performance of these models in classifying reviews into positive or negative sentiments. Performance metrics such as accuracy, precision, recall, F1-score, and the Area Under the Receiver Operating Characteristic Curve (AUC) were utilized to gauge the models' efficacy. The Distil BERT model exhibited the highest accuracy of 96.30%, demonstrating its superior capability in handling sentiment analysis tasks. Future work will focus on addressing the limitations observed in misclassification cases, particularly in detecting sarcasm and handling mixed sentiments.

Contents

1 Introduction 4

2 Related Work 5

3 Exploratory Data Analysis and Preprocessing 7

3.1 Class Distribution 7

3.2 Language Distribution 8

3.3 Data Preprocessing 8

3.4 Impact of Preprocessing 10

3.5 Distribution of Review Lengths 12

3.6 Optimal Vocabulary size 13

4 Methodology 14

4.1 Rationale for Model Selection 15

4.2 LSTM Model Architecture 16

4.3 LSTM-GRU Architecture 18

4.4 CNN Architecture 19

4.5 Distil BERT Architecture 20

5 Experiments 22

5.1 Experimental Setup 22

5.2 Distil BERT 22

5.2.1 Choosing Distil BERT over BERT 23

5.2.2 Model Architecture and Fine-tuning 23

5.2.3 Tokenization with Distil BERT 24

5.2.4 Fine-tuning the Model 25

5.2.5 Challenges Faced 25

5.3 LSTM Model 28

5.3.1 Learning and Accuracy Curves 29

5.4 LSTM+GRU 31

5.4.1 Reason for Adding the GRU Layer 31

5.4.2 Learning and Accuracy Curve Analysis of The LSTM-GRU Model 32

5.5 CNN Model 33

6 Results 35

6.1 Performance Metrics of The LSTM Model 35

6.2 Performance Metrics of the LSTM+GRU Model 37

6.3 Performance Metrics of the CNN Model 38

6.4 Performance Metrics of the Distil BERT Model 40

7 Discussion 42

7.1 Model Selection for Specific Tasks 44

7.2 Discussion on Misclassified Reviews 44

8 Conclusion and future work 46

1 Introduction

Sentiment analysis, a pivotal area within natural language processing (NLP), focuses on discerning the sentiment expressed in textual data [1]. With the proliferation of online reviews and user-generated content, sentiment analysis has gained substantial importance for businesses aiming to understand customer opinions and improve their products and services [2]. This study concentrates on binary sentiment analysis, wherein the objective is to classify text as expressing either positive or negative sentiment. The binary classification approach simplifies the sentiment analysis process and enhances the interpretability of the results, making it particularly valuable for large-scale datasets like those from e-commerce platforms [3].

Amazon, one of the largest e-commerce platforms globally, generates an immense volume of review data daily. Analyzing this data can yield significant insights into consumer satisfaction and product performance [4]. By examining the sentiments expressed in these reviews, businesses can identify strengths and weaknesses in their products and make informed decisions to enhance customer experience. This report presents an in-depth analysis of Amazon review texts using advanced machine learning techniques to achieve accurate sentiment classification. The extensive dataset utilized in this study consists of 3.6 million reviews, providing a robust foundation for training and evaluating various sentiment analysis models [5].

To tackle the sentiment classification task, we employed several neural network models, each offering unique strengths and capabilities in handling textual data. The models evaluated include Convolutional Neural Networks (CNN) [6], Long Short-Term Memory networks (LSTM) [7], a hybrid LSTM-Gated Recurrent Unit (GRU) model [8], and a transformer-based Distil BERT model [9]. CNNs are known for their proficiency in capturing local patterns and dependencies in text, making them suitable for tasks that require quick inference and efficient processing of large datasets [10]. LSTMs, on the other hand, are designed to capture long-term dependencies in sequential data, making them effective for understanding the context over extended text sequences. This capability is particularly useful for sentiment analysis, where the sentiment of a review may be influenced by its overall structure rather than isolated phrases [11].

The hybrid LSTM-GRU model combines the strengths of both LSTM and GRU units. GRUs, or

Gated Recurrent Units are a simplified version of LSTMs that offer comparable performance while being computationally more efficient [8]. By integrating both architectures, the hybrid model aims to leverage detailed context understanding and maintain computational efficiency. This hybrid approach addresses some of the limitations observed in models that rely solely on either LSTM or GRU units, offering a balanced performance across various evaluation metrics. [12].

Distil BERT, a distilled version of the BERT (Bidirectional Encoder Representations from Transformers) model, represents the most advanced architecture evaluated in this study [9]. Distil BERT retains 97% of BERTs performance while being 60% faster and smaller, making it a practical choice for large-scale sentiment analysis tasks. Transformer-based models like Distil BERT are renowned for their ability to understand and represent complex language structures, thanks to their self-attention mechanisms that capture relationships between words in a sentence [13]. This advanced capability makes Distil BERT particularly effective for nuanced sentiment analysis where context and subtleties in language play a crucial role [14].

Through this comparative study, we aim to identify the most effective model for sentiment analysis of Amazon reviews, providing insights into their performance and potential applications. The results highlight the trade-offs between model complexity, training time, and accuracy, offering guidance for future research and practical implementations in the field of sentiment analysis. By systematically evaluating these models, we contribute to the ongoing efforts to enhance sentiment analysis methodologies and apply them effectively in real-world scenarios, ultimately aiding businesses in making data-driven decisions based on customer feedback [15].

2 Related Work

Sentiment analysis has been a popular research topic for many years, with a variety of approaches proposed to tackle the problem. Early methods primarily focused on lexicon-based techniques, where sentiment is determined using pre-defined dictionaries of positive and negative words [16]. While these methods are straightforward, they often fail to capture the nuances of context and sentiment.

Machine learning approaches marked a significant advancement in sentiment analysis. Classical machine learning models such as Support Vector Machines (SVM) and Naive Bayes classifiers have been widely used, relying on manually engineered features like bag-of-words, ngrams, and TF-IDF [17, 18]. These models, although effective to some extent, are limited by their dependence on feature engineering and inability to understand complex language patterns.

With the advent of deep learning, more sophisticated models have been developed. Convolutional Neural Networks (CNN) have been successfully applied to text classification tasks, leveraging their ability to identify local patterns through convolutional filters [6]. Recurrent Neural Networks (RNN) and their variants, such as Long Short-Term Memory (LSTM) networks, address the limitations of fixed-size context windows by capturing sequential dependencies in text data [7].

The introduction of the Transformer architecture brought a paradigm shift in natural language processing. BERT (Bidirectional Encoder Representations from Transformers) represents a significant leap forward, providing deep bidirectional context understanding by processing text in both directions [19]. However, BERT's large model size and computational requirements pose challenges for deployment in resource-constrained environments. To address this, Distil BERT was introduced as a smaller, faster, and lighter alternative, retaining much of BERT's performance while being more efficient [20].

In sentiment analysis, these deep learning models have demonstrated substantial improvements over traditional methods. For instance, the use of CNNs in sentiment classification of movie reviews showed significant performance gains due to their ability to capture hierarchical features [21]. Similarly, RNNs and LSTMs have been effective in understanding the sequential nature of text, leading to better sentiment classification results [22]. More recently, transformerbased models like BERT and Distil BERT have set new benchmarks in various natural language understanding tasks, including sentiment analysis [19, 20].

This project builds on these advancements by comparing the performance of CNNs, RNNs, LSTMs, and Distil BERT in the context of sentiment analysis on a large dataset of Amazon product reviews. The goal is to identify the most effective model for this task, considering both accuracy and computational efficiency.

3 Exploratory Data Analysis and preprocessing

Our journey of sentiment analysis began with exploratory data analysis (EDA) to understand the properties and structure of the dataset [23]. Given the large size of the dataset, initial EDA efforts were challenging due to limited computational resources. To mitigate this, we initially worked with a smaller subset of the data to develop and test our EDA code. Once we had a functional EDA process, we applied it to the entire dataset.

The dataset consisted of 4 million amazon products reviews, with the training set containing 3.6 million reviews and the test set containing 400,000 reviews. The dataset was in a Fast Text file format with reviews and their corresponding labels on each line. In the dataset, negative reviews were given the label '__label__1'and positive reviews were given the label '__label__2'. While loading the dataset and feeding it into the model, we labeled negative reviews as 0 and positive reviews as 1. Additionally, there were no null values in the dataset, which simplified preprocessing.

3.1 Class Distribution

The dataset was evenly split between positive and negative reviews, with 1.8 million reviews for each class in the training set and 200,000 reviews for each class in the test set. This balance ensures that our models do not become biased towards either sentiment.

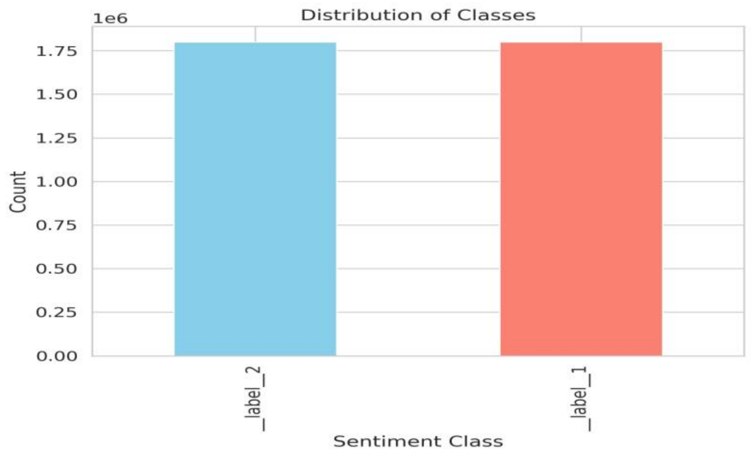


Figure 1: Distribution of sentiment classes in the dataset.

3.2 Language Distribution

We also analyzed the distribution of reviews across different languages to ensure that our dataset predominantly comprised English reviews. This analysis confirmed that the vast majority of reviews were in English, allowing us to proceed without additional multilingual processing.

Among the 3,600,000 reviews, 3,591,085 were in English. There were reviews in 30 different languages, and for 10 reviews, detecting the language was not possible. A few thousand reviews, specifically 6,144, were in Spanish. Besides English and Spanish, reviews in other languages were in the low hundreds, and some languages had only a handful of reviews, countable on one hand. This distribution demonstrates a predominant use of English in the dataset,

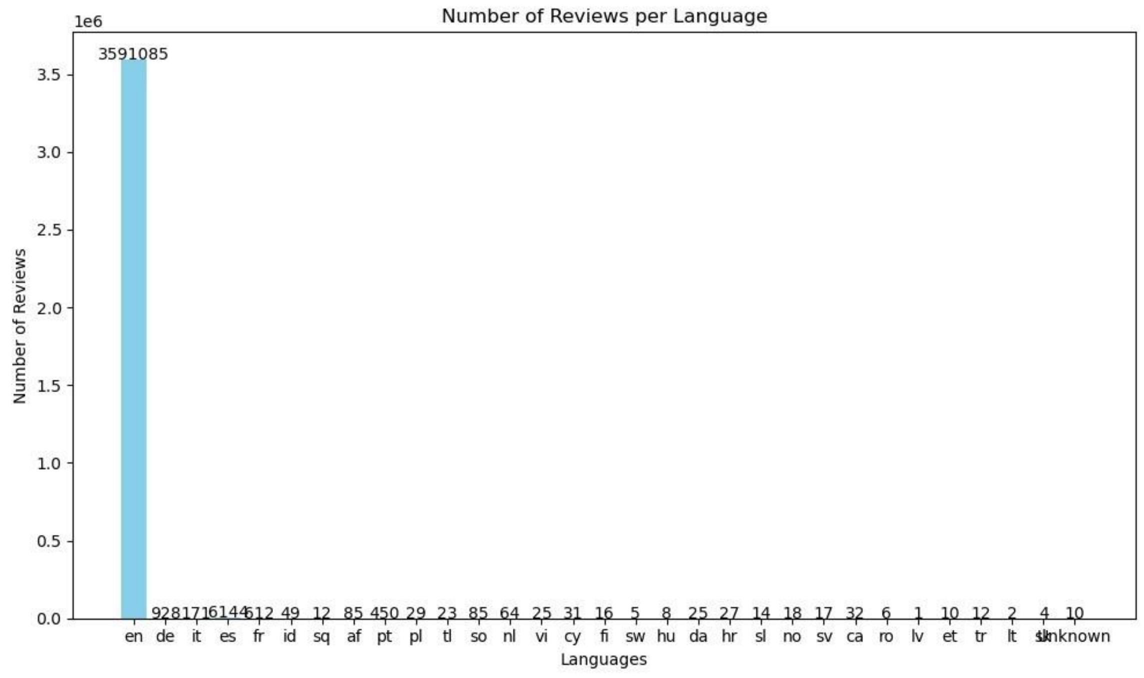


Figure 2: Distribution of languages in the reviews.

minimizing the need for extensive multilingual processing

3.3 Data Preprocessing

Based on the insights from our EDA, we implemented the following preprocessing steps:

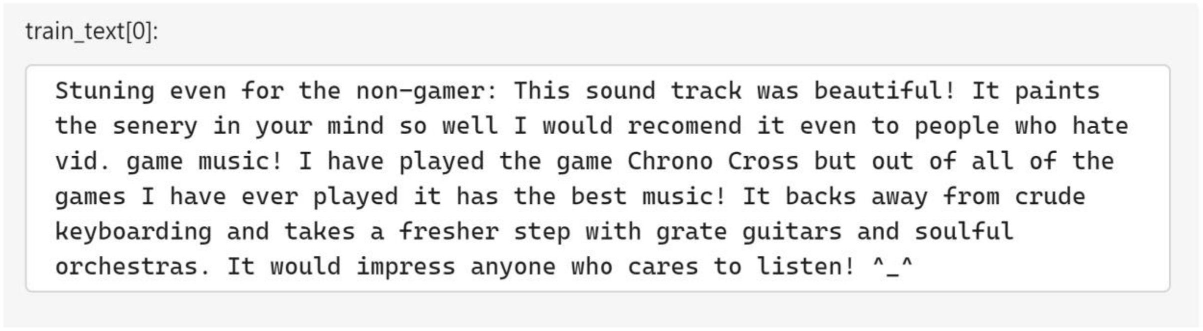
Cleaning and Normalization: The cleaning and normalization section of the preprocessing for

Amazon review text involves several steps aimed at transforming the raw text data into a format that is more suitable for analysis. In the provided code snippet, this process is implemented using regular expressions to remove non-alphanumeric characters, convert text to lowercase, and eliminate non-ASCII characters. First, the text is converted to lowercase to ensure consistency in representation, as uppercase and lowercase versions of words should be treated equally during analysis. This step helps prevent redundancy in the data and ensures that the model can generalize well across different capitalizations of the same word.

Next, non-alphanumeric characters, such as punctuation marks, are replaced with spaces.
This helps in segmenting the text into individual words more effectively and removes unnecessary noise from the data. Punctuation marks often do not carry significant meaning in sentiment analysis and can be safely removed without loss of information.

Finally, non-ASCII characters, which include special characters and emojis, are removed from the text. While these characters may add expressive or visual elements to the text, they can complicate the analysis process and introduce noise. Removing them helps simplify the text data and ensures that the model focuses on the semantic content of the reviews rather than extraneous symbols. Overall, cleaning and normalization are essential preprocessing steps in sentiment analysis of Amazon review text. By standardizing the text data and removing irrelevant characters, these steps help create a cleaner and more consistent dataset, leading to more accurate sentiment analysis results. Additionally, they contribute to improving the generalizability and interpretability of the machine learning models trained on the data.

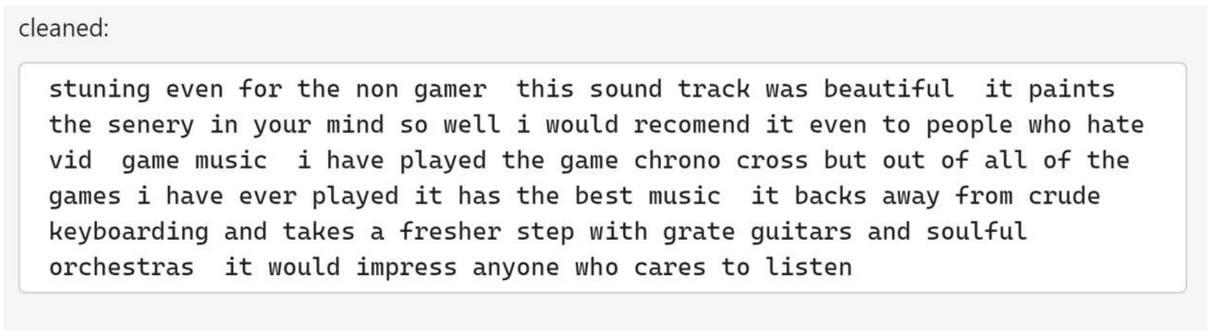
First train text before cleaning and after cleaning are shown below:



train_text[0]:

```
Stuning even for the non-gamer: This sound track was beautiful! It paints
the senery in your mind so well I would recomend it even to people who hate
vid. game music! I have played the game Chrono Cross but out of all of the
games I have ever played it has the best music! It backs away from crude
keyboarding and takes a fresher step with grate guitars and soulful
orchestras. It would impress anyone who cares to listen! ^_^
```

Figure 3: Original Train Text


cleaned:

```
stuning even for the non gamer  this sound track was beautiful  it paints
the senery in your mind so well i would recomend it even to people who hate
vid  game music  i have played the game chrono cross but out of all of the
games i have ever played it has the best music  it backs away from crude
keyboarding and takes a fresher step with grate guitars and soulful
orchestras  it would impress anyone who cares to listen
```

Figure 4: Cleaned and Normalized Train Text

3.4 Impact of Preprocessing

We meticulously tracked the impact of our preprocessing steps, as illustrated in the figures below.

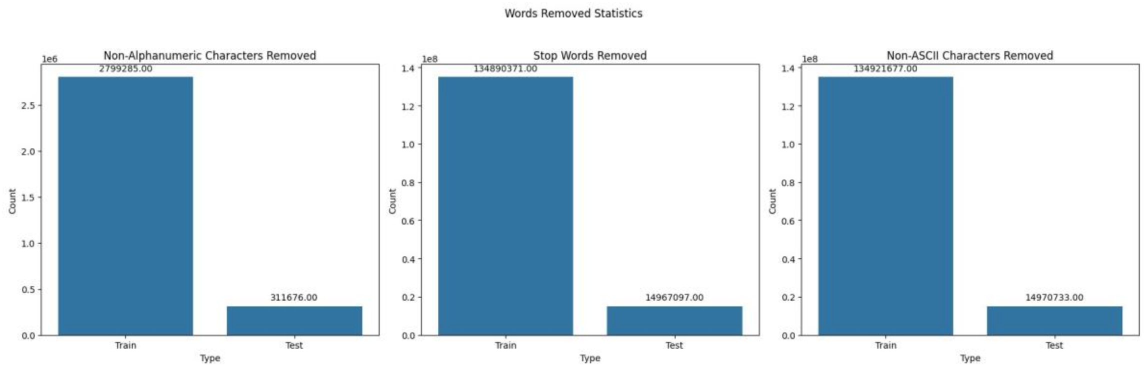


Figure 5: Number of different kinds of words removed during preprocessing.

Figure 5 shows the total number of non-alphanumeric characters, stop words, and non-ASCII characters removed during preprocessing. In the training set, we removed 2,799,285 nonalphanumeric characters, 134,890,371 stop words, and 134,921,677 non-ASCII characters. In the test set, we removed 311,676 non-alphanumeric characters, 14,967,097 stop words, and 14,970,707 non-ASCII characters.

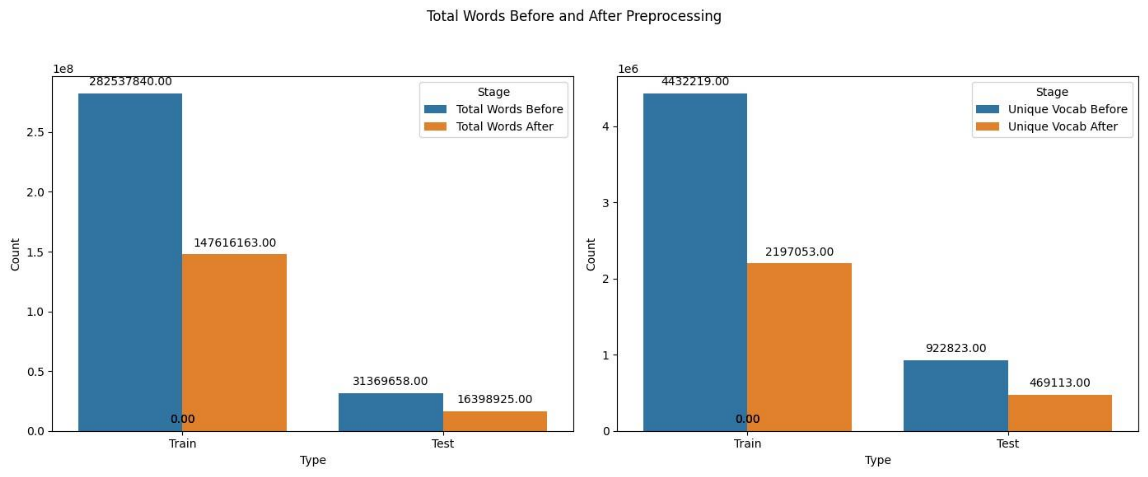


Figure 6: Comparison of vocabulary size before and after preprocessing.

Figure 6 compares the total and unique vocabulary sizes before and after preprocessing. In the training dataset, the total word count was reduced from 282,537,840 to 147,616,163, and the unique vocabulary size was reduced from 4,432,219 to 2,197,053. Similarly, in the test dataset, the total word count was reduced from 31,369,658 to 16,398,925, and the unique vocabulary size was reduced from 922,823 to 469,113.

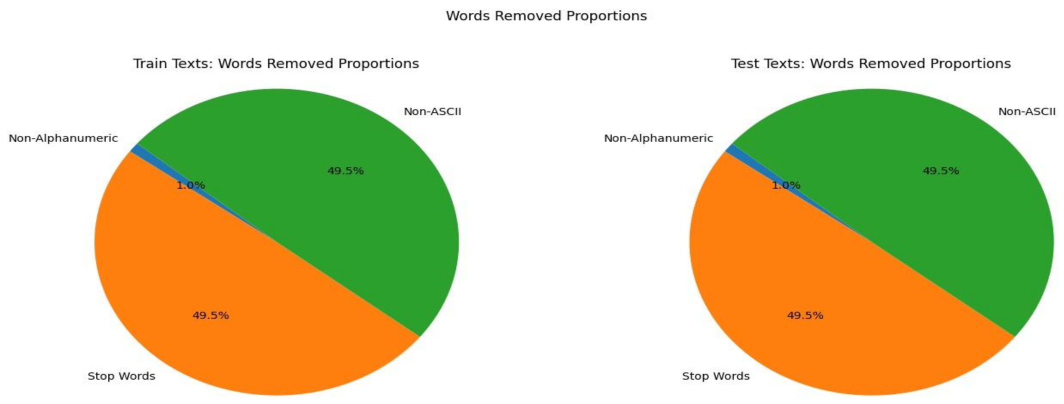


Figure 7: Percentage of different kinds of words removed during preprocessing.

Figure 7 presents a pie chart showing the proportion of different types of words removed during preprocessing. In both the training and test datasets, 49.5% of the removed words were stop words, 49.5% were non-ASCII characters, and 1% were non-alphanumeric characters.

3.5 Distribution of Review Lengths

We analyzed the distribution of review lengths to determine an optimal sequence length for our models. Our analysis showed that review lengths ranged from 3 to 258 words. We also found out that the average length of the review was 78. This variation in length required careful consideration for preprocessing and model input.

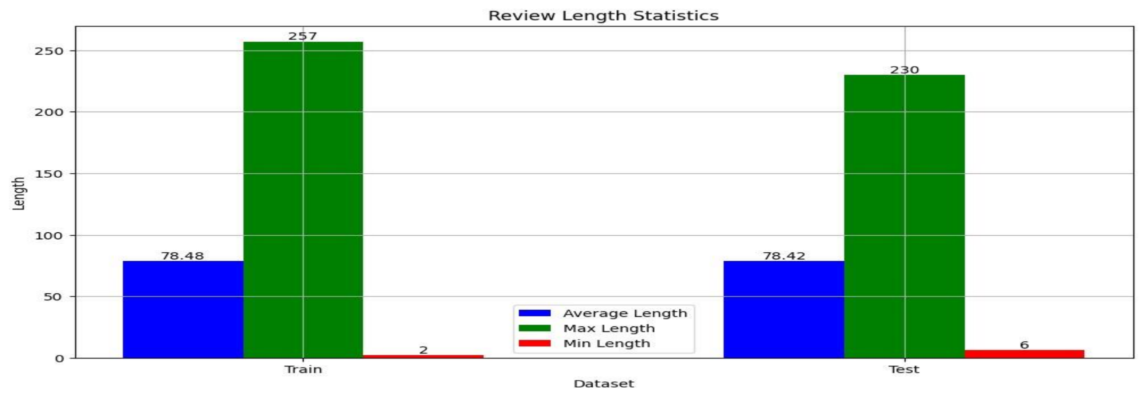
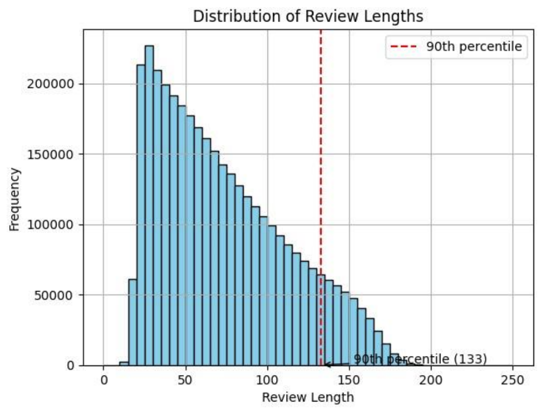
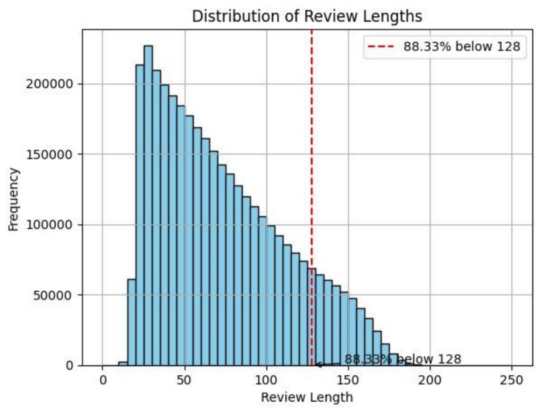


Figure 8: Distribution of review lengths in the dataset.

We found that a length of 133 words would include 90% of the reviews without truncation. However, since computational efficiency is better for dimensions that are powers of two, we chose 128 as our maximum length. This allowed 88% of the reviews to be included without truncation while ensuring computational efficiency.



(a) Distribution of maximum review lengths (90% of text without truncation).



(b) Percentage of reviews within maximum length of 128 characters (without truncation).

Figure 9: Distribution and Percentage of Reviews Based on Maximum Length

3.6 Optimal vocabulary size

Now, after deciding on a maximum length to ensure both accuracy and efficiency, we needed to determine the vocabulary size. A large vocabulary increases the size of the model exponentially. To tackle this, we explored the most frequently repeated vocabulary in the dataset. Plotting the unique vocabulary against cumulative frequency, we observed a plateau after encountering around 14,000 unique words. This suggests that within this vocabulary, we have all the necessary repeated words to capture sentiment while keeping the model lightweight and efficient.

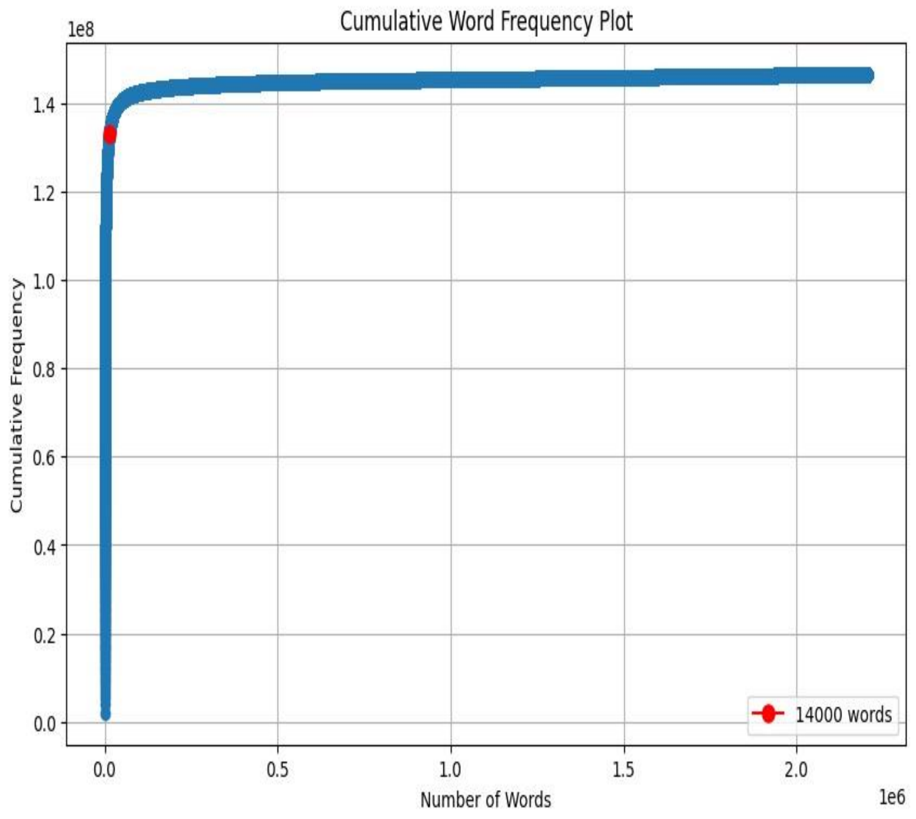


Figure 10: Graph showing frequently repeating words versus cumulative frequency.

4 Methodology

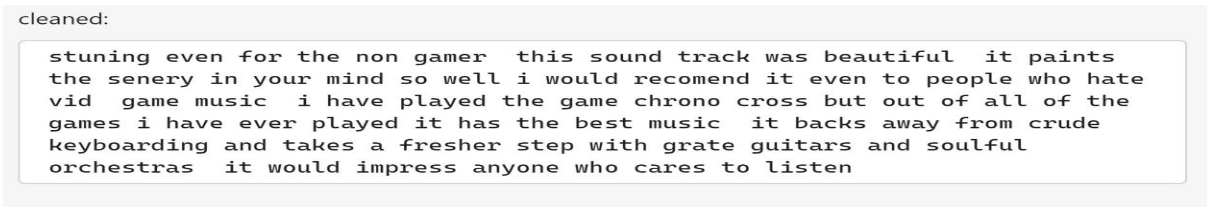
In this section, we outline the methodology adopted for performing binary sentiment analysis on the preprocessed dataset. We had two fast text files, one for training and another for testing. We divided the training dataset into training and validation using sklearn train_test_split function. We divided 80 percentage to training and 20 percentage to validation. To gain a comprehensive understanding of different neural network architectures and their performance on sentiment analysis tasks, we experimented with four distinct models: Convolutional Neural Networks (CNN), Long Short-Term Memory networks (LSTM), a hybrid LSTM+GRU model, and the Transformer-based Distil BERT model.

Prior to model implementation, the dataset underwent extensive preprocessing and Exploratory Data Analysis (EDA). The preprocessing steps included text cleaning, tokenization, and padding sequences to a uniform length. The tokenization and padding section of preprocessing plays a crucial role in preparing text data for machine learning models, particularly in the context of natural language processing tasks like sentiment analysis.

Tokenization involves breaking down the textual data into smaller units, typically words or subworlds, which serve as the fundamental building blocks for analysis. In the provided code snippet, the text is tokenized using a tokenizer object, where each word is assigned a unique numerical identifier. This numerical representation facilitates the model's ability to process and interpret the textual data, as it transforms the text into a format that can be understood and processed by machine learning algorithms.

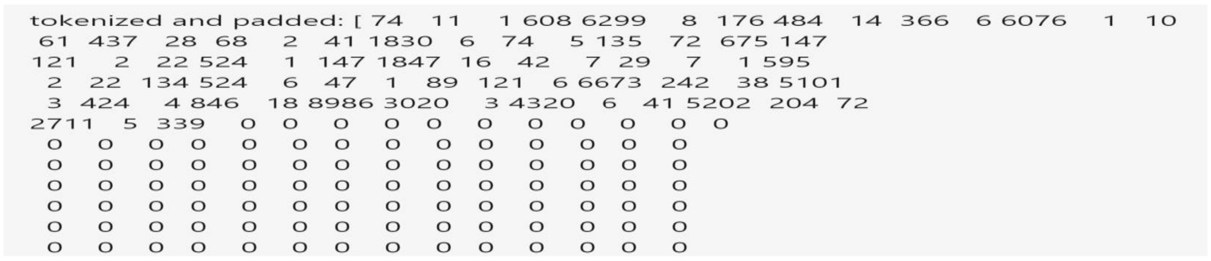
Padding, on the other hand, is essential for ensuring that all sequences of tokenized text have uniform length. In many machine learning models, including neural networks, inputs must be of consistent size to be processed efficiently. By padding sequences with zeros to match the length of the longest sequence, we ensure that all inputs have the same dimensions. This prevents issues like dimension mismatches during training and allows for batch processing of data, which is crucial for optimizing computational efficiency.

Overall, tokenization and padding are necessary preprocessing steps to transform raw textual data into a format suitable for training machine learning models. Tokenization enables the conversion of text into numerical representations, while padding ensures uniformity in input dimensions, thereby facilitating the model's ability to learn meaningful patterns and relationships within the text data. First train text before and after tokenization and padding is shown below:



```
cleaned:
stuning even for the non gamer  this sound track was beautiful  it paints
the senery in your mind so well i would recomend it even to people who hate
vid  game music  i have played the game chrono cross but out of all of the
games i have ever played it has the best music  it backs away from crude
keyboarding and takes a fresher step with grate guitars and soulful
orchestras  it would impress anyone who cares to listen
```

Figure 11: Cleaned Text



```
tokenized and padded: [ 74   11    1 608 6299    8  176 484   14  366   6 6076    1   10
  61  437   28  68    2   41 1830    6   74    5 135   72  675 147
 121    2   22 524    1  147 1847   16   42    7  29    7    1 595
   2   22  134 524    6   47    1   89  121    6 6673  242   38 5101
   3  424    4 846   18 8986 3020    3 4320    6   41 5202  204   72
2711    5  339    0    0    0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0
```

Figure 12: Tokenized and Padded Sequence

EDA provided insights into the dataset’s structure, distribution of sentiment classes, and common patterns within the text data. Visualizations and summary statistics from EDA guided the subsequent modeling efforts.

4.1 Rationale for Model Selection

While it is well-known that advanced models like Distil BERT are highly effective for natural language processing tasks, we chose to experiment with a variety of architectures for several reasons:

- Educational Value: Exploring different architectures provides a deeper understanding of how various neural network designs operate and their respective strengths and weaknesses in handling text data.
- Performance Comparison: By evaluating simpler models such as CNN and LSTM alongside advanced models like Distil BERT, we can track performance improvements and understand the incremental benefits of more recent architectures.
- Model Robustness: Testing multiple models allows us to assess their robustness and adaptability to the specific characteristics of our dataset.

- Historical Perspective: By starting with traditional models and progressing to advanced ones, we can document the evolution of sentiment analysis techniques and the impact of architectural advancements.

4.2 LSTM Model Architecture

The LSTM model for binary sentiment analysis of 3.6 million Amazon review texts is designed to effectively capture the sequential nature of text data. Below is a detailed breakdown of the model architecture:

The architecture chosen for the LSTM model in binary sentiment analysis of Amazon review.

Layer (type)	**Output Shape**	**Param #**	**Description**
input_layer (InputLayer)	(None, 255)	0	Accepts sequences of up to 255 tokens.
embedding_layer (Embedding)	(None, 255, 128)	1,536,000	Transforms the input tokens into dense vectors of 128 dimensions.
spatial_dropout (SpatialDropout1D)	(None, 255, 128)	0	Applies dropout to the embedded sequences for regularization.
bidirectional_lstm (Bidirectional)	(None, 255, 256)	263,168	Bi-directional LSTM layer with 128 units in each direction, capturing context from both past and future.
global_max_pooling (GlobalMaxPooling1D)	(None, 256)	0	Reduces each feature map to its maximum value, reducing the tensor to 256 features.
dense (Dense)	(None, 128)	32,896	Fully connected layer with 128 units, interpreting the extracted features.
dropout (Dropout)	(None, 128)	0	Applies dropout to the dense layer for regularization.
output_layer (Dense)	(None, 1)	129	Dense layer with a single unit and a sigmoid activation function, providing the binary output.
Total		1,832,193	
Trainable params		1,832,193	
Non-trainable params		0	

Table 1: LSTM Model Architecture

texts as shown in Table 1 are specifically designed to leverage the strengths of various neural network components to address the challenges of processing large-scale textual data. Below is a detailed breakdown of the reasons for each component of the architecture:

- Input Layer: The model starts with an input layer that accepts sequences of up to 255 tokens. This length is chosen according to the distribution of review text in the EDA section to capture sufficient context from each review while keeping computational complexity manageable. The sequences represent the text reviews, enabling the model to handle variable-length inputs consistently.
- Embedding Layer: Maps each token to a dense vector of 128 dimensions. The embedding layer is crucial because it allows the model to convert sparse representations of words into dense vectors that capture semantic similarities and relationships between words. This transformation is essential for understanding the nuances of language in the reviews. Embedding layers are known for their effectiveness in capturing semantic information and have been extensively used in natural language processing (NLP) tasks. [24].
- Regularization: Regularization is essential to prevent overfitting, especially when dealing with a large dataset like 3.6 million reviews. Spatial Dropout1D randomly drops entire feature maps instead of individual elements, which helps make the model robust against overfitting and ensures that the learned features are not overly reliant on specific tokens. This technique has been shown to improve model generalization [25].
- LSTM: Long Short-Term Memory (LSTM) networks are effective in handling sequential data due to their ability to retain long-term dependencies. The bidirectional LSTM layer allows the model to consider both past (left context) and future (right context) information for each token, enhancing the understanding of context in a review. This bidirectional approach captures comprehensive context from the text, which is vital for sentiment analysis where the sentiment can be influenced by the entire sentence structure. The effectiveness of bidirectional LSTMs in capturing context has been well-documented [26].
- Global Max Pooling1D: This layer reduces the dimensionality of the data by selecting the maximum value from each feature map. This operation helps in distilling the most

important features from the LSTM output, thereby reducing the data size and highlighting the most relevant information for classification. It ensures that the most significant features detected by the LSTM are passed to the next layers. Global max pooling is a common technique for feature selection in neural networks [27].

- Dense Layer: A fully connected dense layer with 128 neurons and a ReLU activation function is used to learn complex patterns from the pooled features. The ReLU activation helps in capturing non-linear relationships within the data, which is crucial for accurately classifying sentiments that are influenced by subtle and complex language patterns. The use of dense layers with ReLU activation is standard practice in deep learning for capturing complex patterns [28].
- Dropout: Another regularization step is included through a dropout layer. By randomly setting a fraction of the input units to zero during training, the model becomes more generalizable and less likely to overfit the training data. This additional dropout ensures robustness in the final model. Dropout is a widely used regularization technique in neural networks [29].
- Output Layer: The final dense layer with a single neuron and sigmoid activation function produces a probability score between 0 and 1. This output indicates the sentiment (positive or negative) of the review, making it suitable for binary classification tasks. The sigmoid activation is appropriate for binary sentiment analysis, converting the model's output into a probability that can be easily interpreted. The sigmoid function is commonly used in binary classification tasks [30].

4.3 LSTM-GRU Architecture

The LSTM-GRU model is the same as the LSTM model except for the addition of a GRU layer. This addition aims to enhance the model's ability to capture and learn from complex sequential dependencies in the data.

Layer Type	**Output Shape**	**Parameters**	**Description**
Input	(None, 255)	0	Receives the input sequence
Embedding	(None, 255, 128)	1,536,000	Converts words into dense vectors of fixed size
SpatialDropout1D	(None, 255, 128)	0	Randomly sets a fraction of input units to 0 at each update during training time
Bidirectional LSTM	(None, 255, 256)	263,168	Processes the sequence in both forward and backward directions
Bidirectional GRU	(None, 255, 256)	296,448	Adds GRU units to capture dependencies and improve performance
GlobalMaxPooling1D	(None, 256)	0	Downsamples the input representation by taking the maximum value over time steps
Dense	(None, 64)	16,448	Fully connected layer with ReLU activation
Dropout	(None, 64)	0	Prevents overfitting by randomly setting a fraction of input units to 0 at each update during training time
Dense	(None, 1)	65	Produces the final output with sigmoid activation for binary classification
Total Parameters		2,112,129	

Table 2: LSTM-GRU Model Architecture

4.4 CNN Architecture

Convolutional Neural Networks (CNNs) were chosen for sentiment analysis due to their proven effectiveness in capturing local patterns and hierarchical features in text data. CNNs are particularly well-suited for binary sentiment analysis tasks, where the objective is to classify text into positive or negative sentiment categories.

The CNN model architecture used for sentiment analysis is detailed in Table 3.

Table 3: CNN Model Architecture

Layer	Output Shape	Params	Description	Layer Description
Embedding	(seq_len, emb_dim)	#w × emb_dim	Converts input words into dense vectors.	This layer converts input words into dense vectors of fixed size.
Conv1D	(seq_len - fs + 1, nf)	$(fs \times nf) + nf$	Applies convolutional filters to capture local patterns.	This layer applies convolutional filters to capture local patterns in the input sequence.
ReLU	(seq_len - fs + 1, nf)	0	Applies Rectified Linear Unit activation function.	This layer applies Rectified Linear Unit (ReLU) activation function element-wise to the input.
GlobalMa	xPooling1D(nf,)	0	Reduces dimensionality by taking max value across filters.	This layer reduces dimensionality by taking the maximum value across filters.
Dense	(du,)	$(nf \times du) + du$	Fully connected layer with dropout.	This layer is a fully connected (dense) layer with dropout for regularization.
Dropout	(du,)	0	Randomly drops a fraction of input units to prevent overfitting.	This layer randomly drops a fraction of input units to prevent overfitting.
Output	(1,)	$du+1$	Sigmoid activation for binary classification.	This layer applies a sigmoid activation function for binary classification.
Total params: 805377 (3.07 MB)				
Trainable params: 805377 (3.07 MB)				
Non-trainable params: 0 (0.00 Byte)				

4.5 Distil BERT Architecture

Distil BERT is a distilled version of BERT, a transformer-based model that uses self-attention mechanisms to capture relationships between words in a sentence [20]. Distil BERT retains 97% of BERT's performance while being 60% faster and smaller, making it a practical choice for large-scale sentiment analysis. Our Distil BERT model included multiple transformer layers with self-attention and feed-forward networks.

The Distil BERT model, like other transformer-based models, can be used for binary sentiment analysis by leveraging its ability to understand and represent contextual information in text. Here's how it typically works:

1. Input Encoding: The input text, which could be a sentence or a sequence of sentences, is tokenized into subwords or words, and then converted into numerical tokens using a pre-trained vocabulary. This numerical representation is then fed into the model.

2. Embedding Layer: The tokenized input is passed through an embedding layer, where each token is mapped to a high-dimensional vector representation. This embedding layer

Layer	Output Shape	Params	Connected to	Layer Description
input_ids (InputLayer)	(None, 128)	0		Input layer for input_ids.
attention_mask (Input-Layer)	(None, 128)	0		Input layer for attention_mask.
TFSequenceClassificati	on(None, 2)	66955010	input_ids, attention_mask	Distil BERT model for sequence classification.
dense (Dense)	(None, 1)	3	TFSequenceClassificati	onDense layer for classification.
Total params: 66,955,013				
Trainable params: 66,955,013				
Non-trainable params: 0				

Table 4: Distil BERT Model Architecture

captures semantic similarities between words and helps the model understand the meaning of the input text.

3. Transformer Layers: The token embeddings are then processed through multiple transformer layers. Each transformer layer consists of self-attention mechanisms and feedforward neural networks. The self-attention mechanism allows the model to weigh the importance of each word/token based on its context within the sequence. This helps the model capture long-range dependencies and understand the relationships between words in the text.

4. Classification Head: After processing through the transformer layers, the model generates a sequence of contextual embeddings. These embeddings are then pooled to obtain a fixed-size representation of the input text. Finally, a classification head (typically a dense layer with sigmoid activation) is added on top of the pooled embeddings to perform binary sentiment classification. The output of this layer represents the probability of the input text belonging to each sentiment class (positive or negative).

5. Training: During training, the model's parameters (including the transformer layers and the classification head) are fine-tuned using labeled data (sentiment-labeled text). The model learns to minimize a loss function (such as binary cross-entropy loss) by adjusting its parameters to better predict the correct sentiment labels for the training examples.

6. Inference: During inference, the trained model can be used to predict the sentiment of unseen text inputs. The input text is processed through the model, and the output of the classification head represents the predicted probability of each sentiment class. Typically, a threshold (e.g., 0.5) is applied to these probabilities to make the final binary sentiment prediction.

Overall, the Distil BERT model learns to encode the contextual information of input text and leverage it to make accurate predictions about the sentiment expressed in the text. Its ability to capture semantic relationships and context dependencies makes it well-suited for tasks like sentiment analysis.

5 Experiments

In this section, we describe the experimental setup used to train and evaluate the models. This includes details on the hyperparameters, the process of hyperparameter tuning, and the evaluation metrics used to measure the performance of each model.

5.1 Experimental Setup

The experiments were conducted on a machine equipped with an NVIDIA RTX 3060 GPU, 24 GB of RAM, and a Ryzen 7 6800H processor. The models were implemented using the TensorFlow framework. The dataset was split into training, validation, and test sets as described in Section 4.

5.2 Distil BERT

In this section, we detail the experimental setup, focusing on the application of transfer learning using Distil BERT for the sentiment analysis task. We discuss the rationale for selecting Distil BERT over BERT, the methodology for customizing the model for binary classification, the tokenization process, and the challenges encountered during the process.

5.2.1 Choosing Distil BERT over BERT

BERT (Bidirectional Encoder Representations from Transformers) has set new benchmarks in various natural language processing tasks due to its deep bidirectional representations [19]. However, BERT is computationally expensive and requires significant resources for training and inference, making it less suitable for environments with limited computational power.

Distil BERT [20] is a distilled version of BERT that retains 97% of BERT's language understanding capabilities while being 60% faster and 40% smaller. The key advantages of Distil BERT include:

- Efficiency: Faster training and inference times, which are critical given the large size of our dataset.
- Resource Requirements: Lower memory footprint, making it feasible to train on machines with limited resources.
- Performance: Comparable performance to BERT on various NLP benchmarks, ensuring robust results for our sentiment analysis task.

Given these advantages, we selected Distil BERT to balance performance with computational efficiency.

5.2.2 Model Architecture and Fine-tuning

To adapt Distil BERT for the binary classification task of sentiment analysis, we added a classification layer on top of the pre-trained Distil BERT model. The architecture and fine-tuning process are as follows:

Adding a Classification Layer

The pre-trained Distil BERT model outputs a sequence of hidden states for each token in the input text. For classification tasks, we typically use the hidden state corresponding to the [CLS] token, which serves as an aggregate representation of the entire input sequence. We added a fully connected (dense) layer on top of the Distil BERT model, followed by a sigmoid activation function to output the probability of the review being positive. The architecture of our classification model can be summarized as follows:

- Distil BERT Base: Pre-trained Distil BERT model.

- Dense Layer: A fully connected layer with a single output unit.
- Activation Function: Sigmoid activation to produce a probability score for the binary classification.

The architecture can be represented as:

$$\text{output} = \sigma(\mathbf{W} \cdot \text{Distil BERT}_{[\text{CLS}]} + b)$$

where $\mathbf{W}$ and b are the weights and bias of the dense layer, and σ is the sigmoid activation function.

5.2.3 Tokenization with Distil BERT

Tokenization is a crucial preprocessing step for using Distil BERT. The Distil BERT tokenizer handles text tokenization using Word Piece embeddings, which involves:

- Splitting Text: The tokenizer splits the text into subworld units, ensuring that rare words are broken down into more frequent subworlds. This helps in handling out-of-vocabulary. words effectively.
- Adding Special Tokens: The tokenizer adds special tokens, such as [CLS] at the beginning of each sequence and [SEP] at the end, to mark the start and end of the sequence.
- Gene- Generating Token IDs: Each token (including special tokens) is converted into its corresponding token ID based on the tokenizer's vocabulary.
- Attention Masks: The tokenizer generates attention masks to indicate which tokens should be attended to by the model. Padding tokens are ignored by setting their corresponding mask values to zero.

For instance, a sample tokenization process for the text "I love this product!" would produce the following outputs:

- Tokens: [CLS], I, love, this, product, !, [SEP]
- Token IDs: [101], [1045], [2293], [2023], [4031], [999], [102]
- Attention Masks: [1, 1, 1, 1, 1, 1, 1]

These token IDs and attention masks are then used as inputs to the Distil BERT model during training and inference.

5.2.4 Fine-tuning the Model

Fine-tuning involves updating the pre-trained model's weights on our specific dataset. We employed the following procedure:

1. Dataset Preparation: As described in the Data Preprocessing section, we used the cleaned and preprocessed dataset with a maximum sequence length of 128 tokens.
2. Training Configuration: We used the Adam optimizer with a learning rate scheduler.
3. Loss Function: Binary cross-entropy loss was used to optimize the model for the binary classification task.
4. Training Procedure: We trained the model for 60 epochs, with patience of 4 epoch and applying early stopping based on the validation loss to prevent overfitting.

5.2.5 Challenges Faced

Throughout the project, we encountered several challenges:

Computational Constraints

One of the primary difficulties was the computational constraints posed by the large dataset and the need for extensive preprocessing and model training. Despite selecting Distil BERT for its efficiency, training on a dataset of this magnitude required careful resource management. To address this, we initially sampled a subset of the dataset to experiment with preprocessing and model training. This allowed us to iterate quickly and refine our approach before scaling up to the full dataset.

Tokenization Challenges

Tokenization of our extensive dataset presented a significant challenge due to memory limitations. The Distil BERT tokenizer converts text into token IDs and attention masks, which requires substantial memory. When attempting to tokenize the entire dataset at once, the process exhausted the available memory.

Solution: We implemented batch processing for tokenization. By dividing the dataset into smaller batches (e.g., batches of 10,000 reviews), we managed to tokenize the data without exceeding memory limits. This approach allowed us to handle large datasets efficiently by tokenizing them in manageable chunks.

Learning Rate Challenges

Determining the optimal learning rate was another major challenge. Initially, setting a high learning rate (e.g., 0.0001) led to rapid overfitting, while very low learning rates (e.g., 0.00000001) resulted in extremely slow convergence.

Solution: We experimented with various learning rates and implemented a dynamic learning rate scheduler. Our final approach used a scheduler that adjusted the learning rate during training to balance exploration and exploitation:

1. **Initial Experiments**: - High learning rates (> 0.000001) caused overfitting within a few epochs.

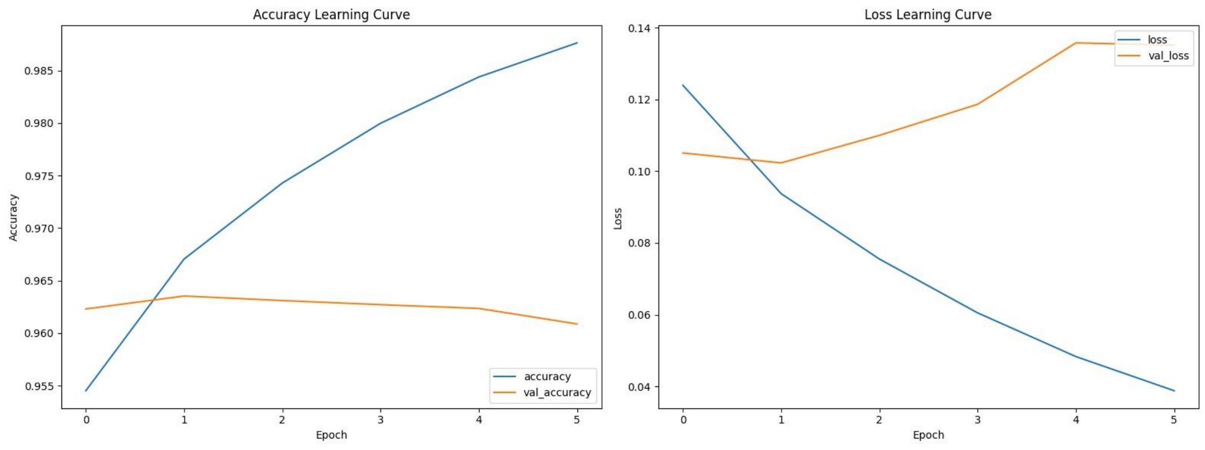
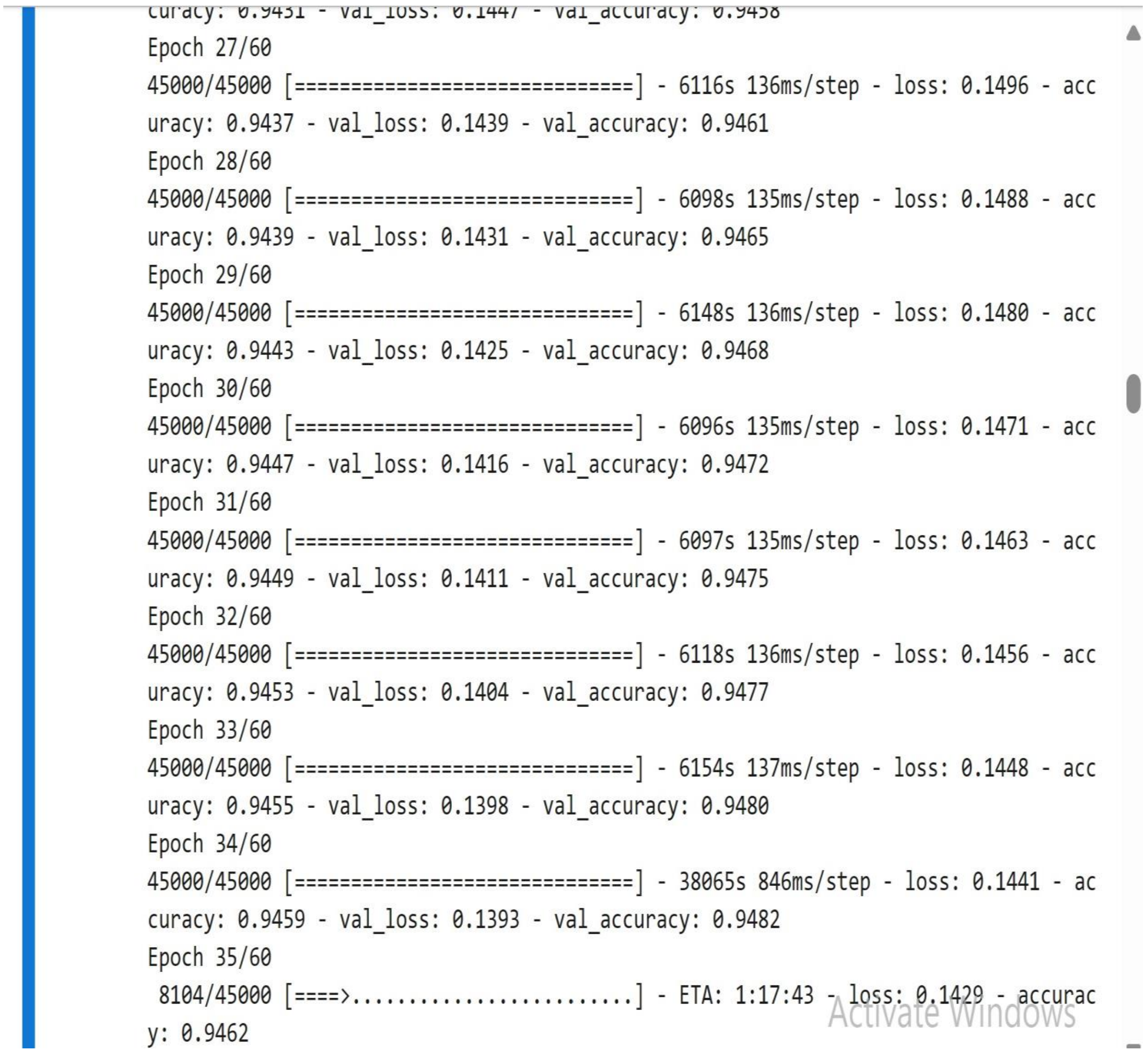


Figure 13: Loss curve during the training of Distil BERT.

- Extremely low learning rates (< 0.0000001) led to impractically slow learning after accuracy reaches over 90%, it was learning at the rate of 0.05 in each epoch. That 60-epoch training worth of 100 hours to train, also could not converge the model.

2. Cosine Annealing: Tried cosine annealing but observed stagnation in learning rate adjustments after several epochs, similar like before.



```
curacy: 0.9431 - val_loss: 0.1447 - val_accuracy: 0.9458
Epoch 27/60
45000/45000 [==============================] - 6116s 136ms/step - loss: 0.1496 - acc
uracy: 0.9437 - val_loss: 0.1439 - val_accuracy: 0.9461
Epoch 28/60
45000/45000 [==============================] - 6098s 135ms/step - loss: 0.1488 - acc
uracy: 0.9439 - val_loss: 0.1431 - val_accuracy: 0.9465
Epoch 29/60
45000/45000 [==============================] - 6148s 136ms/step - loss: 0.1480 - acc
uracy: 0.9443 - val_loss: 0.1425 - val_accuracy: 0.9468
Epoch 30/60
45000/45000 [==============================] - 6096s 135ms/step - loss: 0.1471 - acc
uracy: 0.9447 - val_loss: 0.1416 - val_accuracy: 0.9472
Epoch 31/60
45000/45000 [==============================] - 6097s 135ms/step - loss: 0.1463 - acc
uracy: 0.9449 - val_loss: 0.1411 - val_accuracy: 0.9475
Epoch 32/60
45000/45000 [==============================] - 6118s 136ms/step - loss: 0.1456 - acc
uracy: 0.9453 - val_loss: 0.1404 - val_accuracy: 0.9477
Epoch 33/60
45000/45000 [==============================] - 6154s 137ms/step - loss: 0.1448 - acc
uracy: 0.9455 - val_loss: 0.1398 - val_accuracy: 0.9480
Epoch 34/60
45000/45000 [==============================] - 38065s 846ms/step - loss: 0.1441 - ac
curacy: 0.9459 - val_loss: 0.1393 - val_accuracy: 0.9482
Epoch 35/60
 8104/45000 [====>.........................] - ETA: 1:17:43 - loss: 0.1429 - accurac
y: 0.9462
```

Figure 14: Learning curve showing slow rate of learning after 92%.

2. **Custom Learning Rate Scheduler**: - Developed a custom learning rate schedule to start with a very small learning rate for initial exploration and gradually increased it in stages to accelerate convergence. This scheduler allowed the model to explore various minima during initial epochs and later fine-tune at a more optimal learning rate. By dynamically adjusting the learning rate, we balanced fast initial convergence with the precision required for fine-tuning.

These strategies enabled us to train the model effectively within the constraints of our available resources, ensuring both efficiency and high performance in the sentiment analysis task.

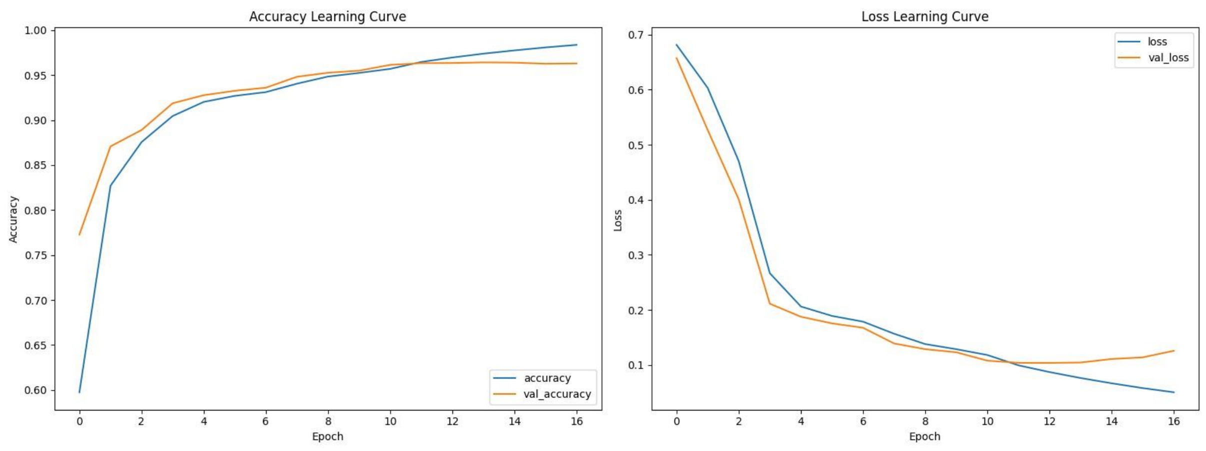


Figure 15: Training Loss and Accuracy

By addressing these challenges, we successfully fine-tuned the Distil BERT model for our sentiment analysis task, achieving high performance and demonstrating the effectiveness of transfer learning in this context.

5.3 LSTM Model

The following hyperparameters were chosen for the experiment:

Hyperparameter	Value
Maximum Sequence Length	255
Vocabulary Size	12,000
Embedding Dimension	128
LSTM Units	128
Dropout Rate	0.1
Batch Size	128
Number of Epochs	60
Optimizer	RMSprop
Learning Rate	0.000025
Loss Function	Binary Cross-Entropy
Metrics	Accuracy, Precision, Recall
Early Stopping Patience	3

Table 5: LSTM Hyperparameters

The chosen hyperparameters (maximum sequence length, vocabulary size, embedding dimension, LSTM units, dropout rate, batch size, number of epochs, optimizer, learning rate, and loss function) are tailored to balance model complexity and computational efficiency. The hyperparameters ensure that the model can effectively learn from a large dataset without overfitting and that it can converge to an optimal solution within a reasonable number of epochs.

The model is compiled with the RMSprop optimizer, using binary cross-entropy as the loss function, and tracks accuracy, precision, and recall as metrics. Early stopping is employed to prevent overfitting, and halting training if the validation loss does not improve for a specified number of epochs.

The training process included callbacks for early stopping and model checkpointing, ensuring the best model is saved based on validation loss. The learning curves for loss and accuracy indicated effective learning and convergence, with minimal overfitting.

The overall architecture combines embedding layers to convert text to dense vectors, bidirectional LSTM to capture long-term dependencies and context from both directions, pooling and dropout layers for regularization, and dense layers for learning complex patterns. This structure is well-suited for the sentiment analysis task, as it can effectively process and learn from the large-scale textual data of Amazon reviews, providing high accuracy, precision, and recall on the validation set.

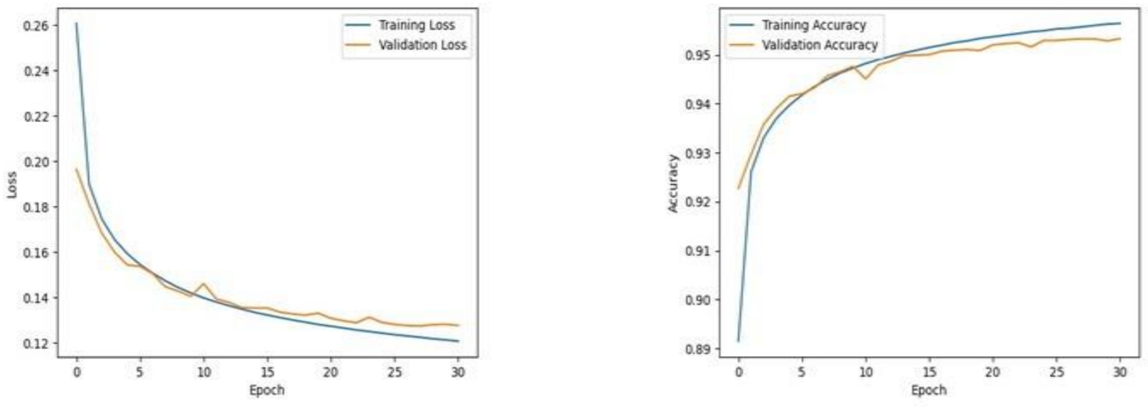
The architecture and hyperparameters are designed to leverage the strengths of deep learning in understanding and classifying text data, making it a robust choice for sentiment analysis in this context. By capturing semantic relationships, context, and key features of the reviews while preventing overfitting, the model is able to generalize well to new, unseen reviews, making it highly effective for sentiment analysis on large datasets like Amazon reviews.

5.3.1 The LSTM Learning and Accuracy Curves

The learning curve, which plots loss against epochs, shows a clear trend of decreasing loss over time for both training and validation datasets. Initially, both training and validation losses start at relatively high values. The training loss drops sharply in the first few epochs, indicating that the model is quickly learning to minimize the error in the training data. The validation loss follows a similar trend, decreasing significantly during the early stages of training, suggesting the model is also improving its performance on unseen data. As training progresses, both curves continue to decline, though the rate of decrease slows down. Around the 10th epoch, the validation loss shows some minor fluctuations, but overall, it maintains a downward trend. The observed fluctuations in the loss are a result of the dropout regularization [25], the nature of the

RMSprop optimizer [29], and the inherent variability in the dataset. These factors combined can cause temporary instability, but as long as the overall trend is improving and the final convergence is stable, the model is performing well. By the 30th epoch, both training and validation losses converge and stabilize, with the training loss slightly lower than the validation loss. This convergence and stabilization indicate a well-trained model that has learned to perform the task with minimal error.

The accuracy curves, which plot accuracy against epochs, complement the learning curves by showing the model's ability to correctly predict sentiments. Initially, the training accuracy starts lower than the validation accuracy, a common occurrence in the early epochs as the model has not yet learned effectively from the training data.



(a) Loss Curve (b) Accuracy Curve

Figure 16: Loss and Accuracy Curves

However, after the first few epochs, both training and validation accuracies increase rapidly. By around the 5th epoch, the training accuracy surpasses the validation accuracy, and both curves continue to rise steadily. This suggests that the model is becoming better at distinguishing between positive and negative sentiments in the reviews.

As the epochs progress beyond the 10th, the validation accuracy exhibits slight fluctuations but generally follows the training accuracy closely. Both curves plateau near the end, with the training accuracy reaching slightly above 95% and the validation accuracy stabilizing just below it. This high level of accuracy indicates that the model is highly effective in predicting sentiments and that there is a good balance between fitting the training data and generalizing it to validation data.

Overall, the provided learning and accuracy curves suggest that the LSTM model is well-trained and effective for sentiment analysis on the given dataset. The model demonstrates significant improvement in both loss and accuracy over the training epochs, with minimal signs of overfitting. The close alignment between training and validation metrics further supports the model's robustness and generalization capability.

5.4 LSTM+GRU

Following hyperparameters were explored while experimenting with the LSTM+GRU model. Hyperparameters of LSTM-GRU Model

5.4.1 Reason for Adding the GRU Layer

The GRU (Gated Recurrent Unit) layer is added to the previous LSTM model to enhance the model's ability to capture and learn from complex sequential dependencies in the data. While LSTM layers are effective at handling long-term dependencies due to their gating mechanisms, GRU layers offer a simpler and often more computationally efficient alternative. The combination of LSTM and GRU layers leverages the strengths of both architectures, potentially leading to better performance and faster training times. By incorporating GRU layers, the model

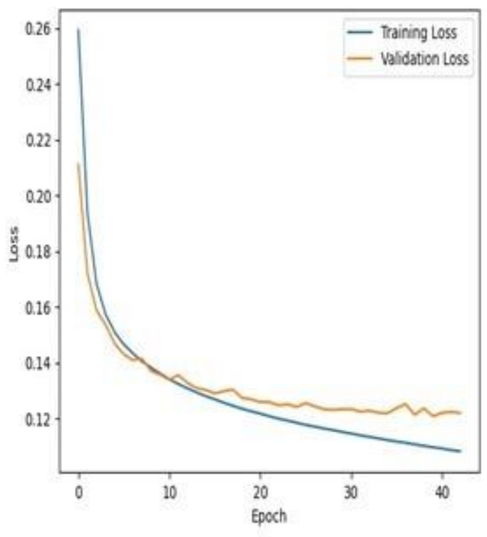
Hyperparameter	Value
Maximum Sequence Length	255
Vocabulary Size	12,000
Embedding Dimension	128
LSTM Units	128
Dropout Rate	0.1
Batch Size	128
Number of Epochs	60
Optimizer	RMSprop
Learning Rate	0.000025
Loss Function	Binary Cross-Entropy
Metrics	Accuracy, Precision, Recall
EarlyStopping Patience	3

Table 6: Hyperparameters of LSTM-GRU Model

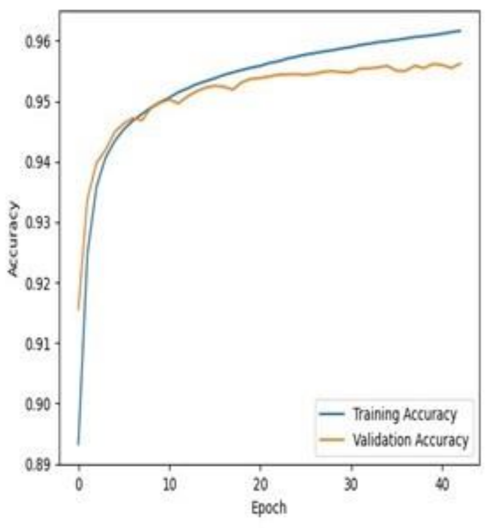
can benefit from the different ways these units process temporal information, thus improving its overall capability in sequential tasks.

5.4.2 The LSTM-GRU Learning and Accuracy Curve

The provided learning and accuracy curves as shown in Figure 3 detail the performance of an LSTM-GRU model. The dataset was divided into training (80%) and validation (20%) subsets, providing a comprehensive evaluation of the model's generalization capabilities.



(a) Loss Curve for LSTM-GRU



(b) Accuracy Curve for LSTM-GRU

Figure 17: LSTM-GRU Learning and Accuracy Curves

The learning curve illustrates the loss over 45 epochs for both the training and validation sets. Initially, the training loss starts at around 0.26 and experiences a steep decline, which indicates that the model is effectively learning the patterns in the data. By around epoch 10, the loss for both training and validation stabilizes, suggesting the model is converging. The training loss continues to gradually decrease, eventually reaching approximately 0.11, while the validation loss remains slightly higher but parallel, indicating good generalization without significant overfitting.

Conversely, the accuracy curve provides insights into the model's performance in terms of correctly predicting the sentiment of the reviews. Starting at a lower accuracy, the model quickly improves, with both training and validation accuracy following a steep upward trend initially.

By around epoch 10, the accuracy begins to plateau, signifying that further epochs contribute marginally to performance improvement. The training accuracy eventually reaches about 96%, while the validation accuracy settles around 95%, indicating the model's strong performance and its robustness across both datasets.

Overall, these curves suggest that the LSTM-GRU model is effectively learning from the training data and generalizing well to the validation data. The minimal gap between the training and validation metrics points towards a well-tuned model with low overfitting, making it a reliable tool for sentiment analysis of Amazon reviews.

5.5 CNN Model

As discussed in the methodology section, the CNN (Convolutional Neural Network) architecture plays a crucial role in sentiment analysis tasks. It excels at capturing local patterns in text data, making it particularly effective for analyzing sequences such as sentences or documents. For our experiment, we implemented a CNN architecture consisting of multiple convolutional layers followed by pooling layers and dense layers for classification. The specific architecture details, including the number of layers, filter sizes, and activation functions, are detailed in the methodology section. Learning Rate and Model Convergence

Selecting the optimal learning rate was crucial to ensure the model converged effectively without overfitting or underfitting. We experimented with various learning rates and found that a learning rate of 0.00001 using the Adam optimizer provided the best results. This balance allowed the model to learn efficiently over multiple epochs.

Hyperparameter Tuning

Fine-tuning hyperparameters such as batch size, number of filters, and kernel sizes was necessary to optimize model performance. Through systematic experimentation and incremental adjustments, we identified the optimal hyperparameters. For instance, a batch size of 256 offered a good trade-off between training stability and resource utilization. Additionally, we adjusted the number of filters and kernel sizes to enhance feature extraction capabilities.

Incremental Improvements

We tracked the performance of our model across different hyperparameter settings using graphs. We started our experiment using adam optimizer and selecting leraning rate of 0.0001. These resulted in quick overfitting and model converges sub optimally within just 6 epochs. We experimented with different learning rates and finally found the optimal learning rate of 0.00001 where the model converges optimally without overfitting.

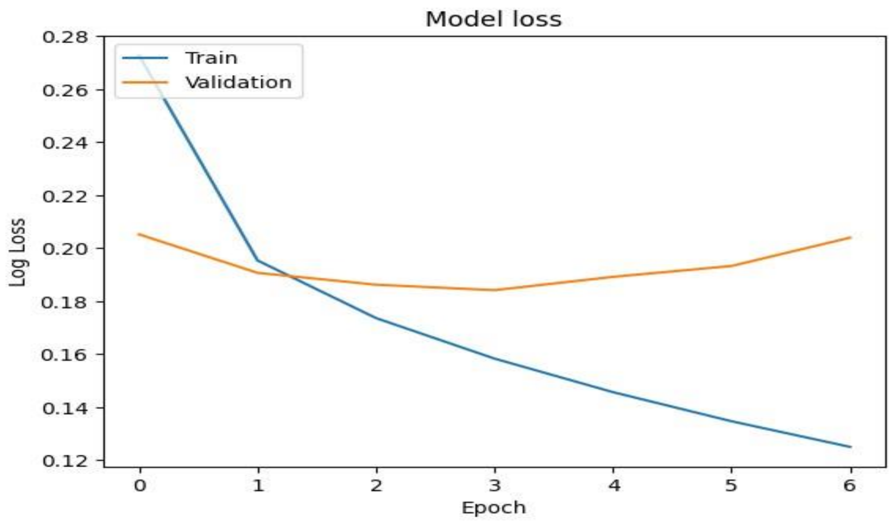


Figure 18: Overfitting of CNN model with learning rate 0.0001

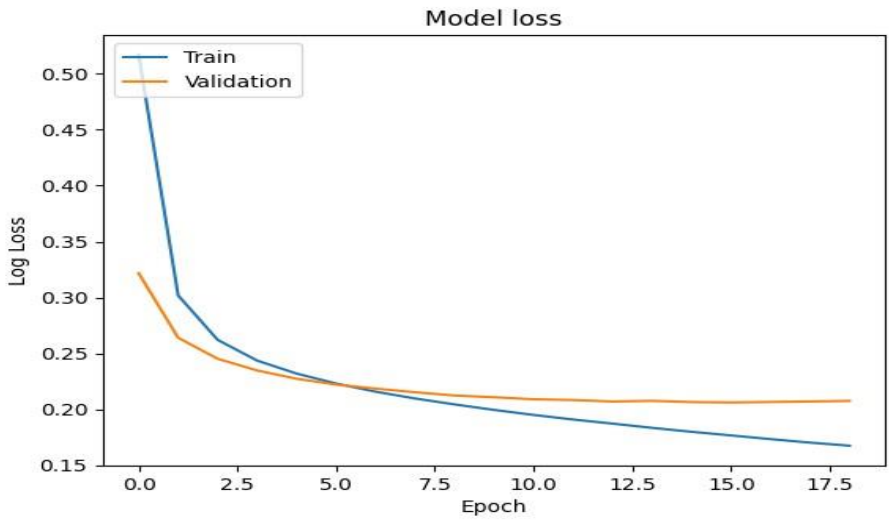


Figure 19: Optimal learning

By methodically tuning hyperparameters and addressing challenges such as learning rate selection, we incrementally improved our model's performance. These optimizations resulted in a CNN model that performed effectively for the sentiment analysis task. Detailed results of these improvements are discussed in the Results section.

6 Results

In this section, we present the results of our experiments. The performance of each model was evaluated on the test set using the metrics defined in Section 5. We include quantitative results in the form of tables and visualizations to provide a comprehensive comparison.

6.1 Performance Metrics of The LSTM Model

The performance metrics of the LSTM model on the test dataset of 400,000 Amazon review texts are shown in Table 7:

Metric	Value
Accuracy	95.33%
Precision	95.73%
Recall	94.89%
F1-Score	95.31%
Sensitivity	94.89%
AUC	0.99

Table 7: LSTM Metrics

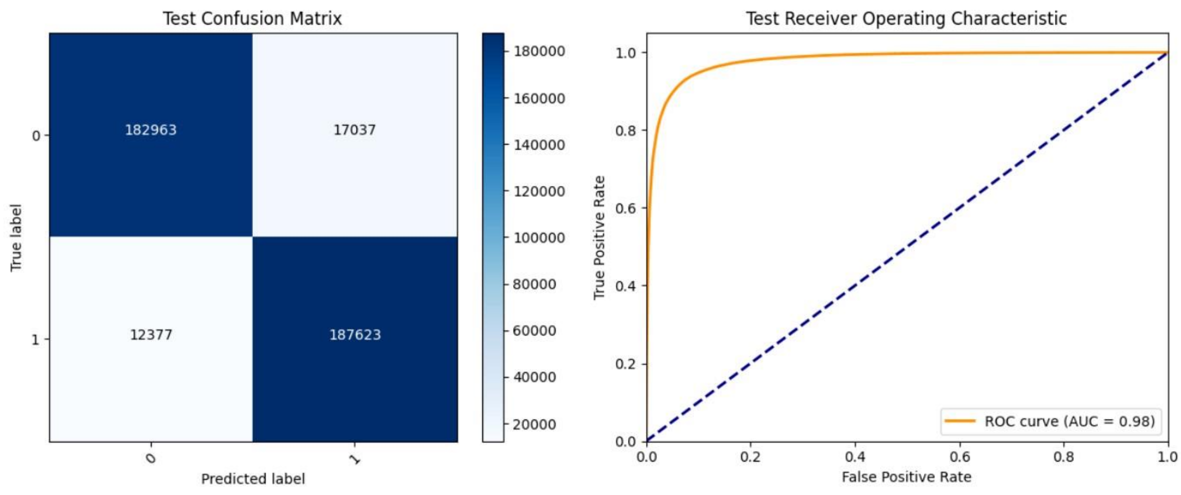


Figure 20: Confusion Matrix and ROC Curve of LSTM Model

The performance of the LSTM model on the dataset, with 400,000 set aside for testing, shows robust metrics across various evaluation parameters. The model achieved an accuracy of 95.33%, indicating that it correctly classified 95.33% of the review texts. This high accuracy is a strong performance indicator, though it does not distinguish between different types of errors, making it essential to look at other metrics for a comprehensive evaluation.

Precision for the LSTM model is 95.73%, which measures the proportion of positive identifications that were actually correct. This high precision value indicates that the model makes very few mistakes in predicting positive sentiment, which is crucial for ensuring that positive sentiment predictions are trustworthy and minimizing false positives.

Recall, on the other hand, is at 94.89%. Recall measures the proportion of actual positives that were correctly identified, showing the model's effectiveness at identifying true positive reviews. The high recall is vital when the goal is to capture as many positive sentiments as possible, which is essential for a comprehensive sentiment analysis.

The F1-Score, which is the harmonic mean of precision and recall, stands at 95.31%. This balanced measure indicates that the model maintains a good equilibrium between precision and recall. A high F1 score reflects robust overall performance, ensuring that the model is both precise and comprehensive in its sentiment classification.

Sensitivity, synonymous with recall, confirms that 94.89% of actual positive cases were correctly identified by the model, reinforcing its effectiveness in capturing true positives. Additionally, the AUC (Area Under the ROC Curve) as shown in Figure 20 is 0.99, signifying excellent model performance in distinguishing between positive and negative classes. A high AUC indicates that the model is highly capable of differentiating between positive and negative sentiments, reducing the chances of classification errors.

The confusion matrix in Figure 20 reveals that the model correctly identified 187,623 positive reviews (true positives) and 182,963 negative reviews (true negatives). It misclassified 17,037 negative reviews as positive (false positives) and 12,377 positive reviews as negative (false negatives). The relatively low number of false positives and false negatives indicates strong model performance.

Overall, the LSTM model demonstrates strong performance metrics with high accuracy, precision, recall, and F1-score, supported by a robust AUC. The confusion matrix reveals a balanced ability to correctly identify both positive and negative sentiments, though there are still some misclassifications. The high precision and recall values suggest that the model is both accurate and comprehensive in its sentiment analysis, making it a reliable tool for analyzing

Amazon review texts. Future improvements could focus on reducing the false positive and false negative rates to enhance the model's reliability even further.

6.2 Performance Metrics of the LSTM+GRU Model

The performance metrics for the LSTM+GRU model on the test dataset of 400,000 Amazon review texts are as follows:

Metric	Value
Accuracy	95.61%
Precision	95.60%
Recall	95.63%
F1-Score	95.62%
Sensitivity	95.63%
AUC	0.99

Table 8: LSTM+GRU Metrics

The LSTM+GRU model demonstrates a high level of performance on the sentiment analysis of the Amazon review texts. The model achieves an accuracy of 95.61%, indicating that it

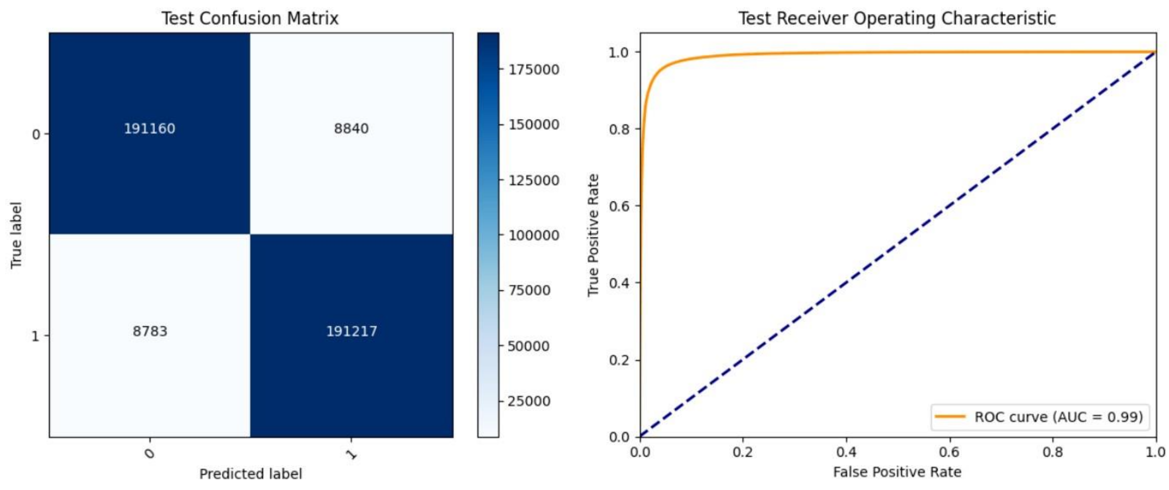


Figure 21: Confusion Matrix and ROC Curve of LSTM+GRU Model

correctly classified a high percentage of reviews. This metric alone suggests a reliable performance, but it must be complemented with other metrics for a comprehensive evaluation.

The precision of the LSTM+GRU model is 95.60%, which measures the proportion of positive identifications that were actually correct. This high precision ensures that the model makes very few false positive errors, making it trustworthy for predicting positive sentiment.

The recall is 95.63%, indicating the model's effectiveness in identifying actual positive reviews. High recall is crucial for sentiment analysis, especially when it is important to capture as many positive sentiments as possible.

The F1-Score, which is the harmonic mean of precision and recall, is 95.62%. This balanced measure reflects the model's ability to maintain a good balance between precision and recall, ensuring both accurate and comprehensive sentiment classification.

The Sensitivity, synonymous with recall, is also 95.63%, reinforcing the model's effectiveness in capturing true positives. The AUC (Area Under the ROC Curve) in Figure 21 is 0.99, signifying excellent performance in distinguishing between positive and negative classes. A high AUC value indicates that the model is highly capable of differentiating between positive and negative sentiments.

The confusion matrix in Figure 21 reveals that the model correctly identified 191,217 positive reviews (true positives) and 191,160 negative reviews (true negatives). It misclassified 8,840 negative reviews as positive (false positives) and 8,783 positive reviews as negative (false negatives). The relatively low number of false positives and false negatives indicates strong model performance.

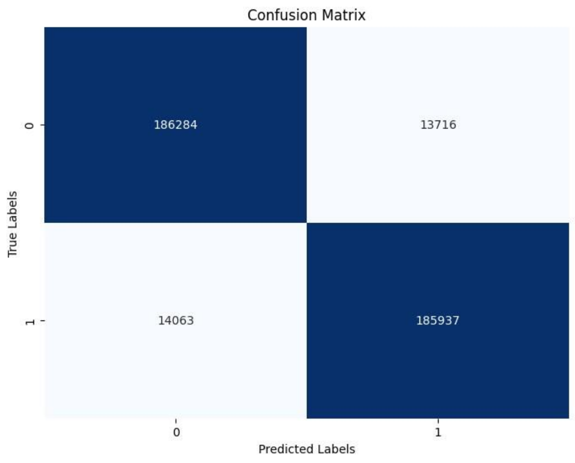
Overall, the LSTM+GRU model exhibits strong performance metrics with high accuracy, precision, recall, and F1-score, supported by a robust AUC. The confusion matrix shows a balanced ability to correctly identify both positive and negative sentiments, despite some misclassifications. The high precision and recall values suggest that the model is both accurate and comprehensive in its sentiment analysis, making it a reliable tool for analyzing Amazon review texts. Future improvements could focus on further reducing the false positive and false negative rates to enhance the model's reliability even further.

6.3 Performance Metrics of the CNN Model

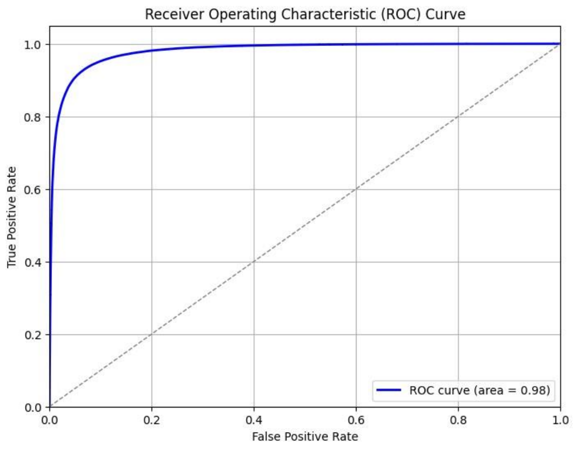
The performance metrics for the CNN model on the test dataset of 400,000 Amazon review texts are as follows:

Metric	Value
Accuracy	93.12%
Precision	93.00%
Recall	93.00%
F1-Score	93.00%
AUC	0.98

Table 9: CNN Metrics



(a) Confusion Matrix and Classification Report



(b) ROC Curve

Figure 22: Confusion Matrix, Classification Report, and ROC Curve of CNN Model

The CNN model demonstrates a high level of performance on the sentiment analysis of the Amazon review texts. The model achieves an accuracy of 93.12%, indicating that it correctly classified a high percentage of reviews. This metric alone suggests a reliable performance, but it must be complemented with other metrics for a comprehensive evaluation.

The precision of the CNN model is 93.00%, which measures the proportion of positive identifications that were actually correct. This high precision ensures that the model makes very few false positive errors, making it trustworthy for predicting positive sentiment.

The recall is 93.00%, indicating the model's effectiveness in identifying actual positive reviews. High recall is crucial for sentiment analysis, especially when it is important to capture as many positive sentiments as possible.

The F1-Score, which is the harmonic mean of precision and recall, is 93.00%. This balanced measure reflects the model's ability to maintain a good balance between precision and recall, ensuring both accurate and comprehensive sentiment classification.

The AUC (Area Under the ROC Curve) in Figure 22 is 0.98, signifying excellent performance in distinguishing between positive and negative classes. A high AUC value indicates that the model is highly capable of differentiating between positive and negative sentiments.

The confusion matrix in Figure 22 reveals that the model correctly identified 185,937 positive reviews (true positives) and 186,284 negative reviews (true negatives). It misclassified 13,716 negative reviews as positive (false positives) and 14,063 positive reviews as negative (false negatives). The relatively low number of false positives and false negatives indicates strong model performance.

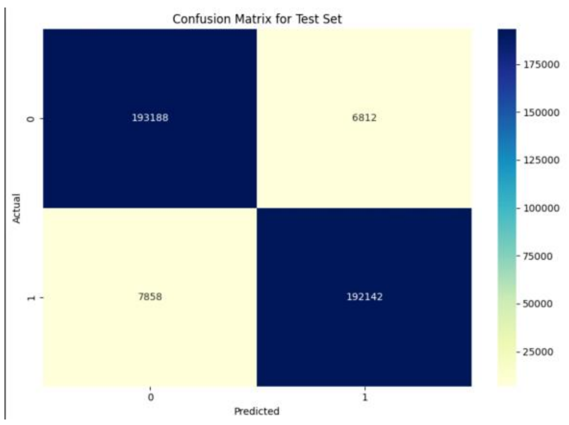
Overall, the CNN model exhibits strong performance metrics with high accuracy, precision, recall, and F1-score, supported by a robust AUC. The confusion matrix shows a balanced ability to correctly identify both positive and negative sentiments, despite some misclassifications. The high precision and recall values suggest that the model is both accurate and comprehensive in its sentiment analysis, making it a reliable tool for analyzing Amazon review texts. Future improvements could focus on further reducing the false positive and false negative rates to enhance the model's reliability even further.

6.4 Performance Metrics of the Distil BERT Model

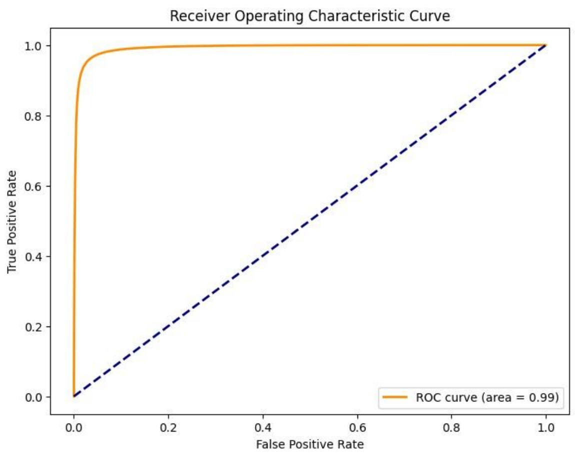
The performance metrics for the Distil BERT model on the test dataset of 400,000 Amazon review texts are as follows:

Metric	Value
Accuracy	96.30%
Precision	97.00%
Recall	96.00%
F1-Score	96.00%
AUC	0.99

Table 10: Distil BERT Metrics



(a) Confusion Matrix and Classification Report



(b) ROC Curve

Figure 23: Confusion Matrix, Classification Report, and ROC Curve of Distil BERT Model

The Distil BERT model demonstrates a high level of performance on the sentiment analysis of the Amazon review texts. The model achieves an accuracy of 96.30%, indicating that it correctly classified a high percentage of reviews. This metric alone suggests a reliable performance, but it must be complemented with other metrics for a comprehensive evaluation.

The precision of the Distil BERT model is 97.00%, which measures the proportion of positive identifications that were actually correct. This high precision ensures that the model makes very few false positive errors, making it trustworthy for predicting positive sentiment.

The recall is 96.00%, indicating the model's effectiveness in identifying actual positive reviews. High recall is crucial for sentiment analysis, especially when it is important to capture as many positive sentiments as possible.

The F1-Score, which is the harmonic mean of precision and recall, is 96.00%. This balanced measure reflects the model's ability to maintain a good balance between precision and recall, ensuring both accurate and comprehensive sentiment classification.

The AUC (Area Under the ROC Curve) in Figure 23 is 0.99, signifying excellent performance in distinguishing between positive and negative classes. A high AUC value indicates that the model is highly capable of differentiating between positive and negative sentiments.

The confusion matrix in Figure 23 reveals that the model correctly identified 192,142 positive reviews (true positives) and 193,188 negative reviews (true negatives). It misclassified 6,812 negative reviews as positive (false positives) and 7,858 positive reviews as negative (false negatives). The relatively low number of false positives and false negatives indicates strong model performance.

Overall, the Distil BERT model exhibits outstanding performance metrics with high accuracy, precision, recall, and F1-score, supported by a robust AUC. The confusion matrix shows a balanced ability to correctly identify both positive and negative sentiments, despite some misclassifications. The high precision and recall values suggest that the model is both accurate and comprehensive in its sentiment analysis, making it a reliable tool for analyzing Amazon review texts. Future improvements could focus on further reducing the false positive and false negative rates to enhance the model's reliability even further.

7 Discussion

The results of our four models: LSTM, LSTM+GRU, CNN, and Distil BERT, on the sentiment analysis task. The comparative performance of the model is shown in th below in the table: 11.

Model	Accuracy	Precision	Recall	F1-Score
LSTM	95.33%	95.73%	94.89%	95.31%
LSTM+GRU	95.61%	95.60%	95.63%	95.62%
CNN	93.12%	93.00%	93.00%	93.00%
Distil BERT	96.30%	97.00%	96.00%	96.00%

Table 11: Performance of different models on the sentiment analysis task.

The results in Table 11 highlight the strengths and limitations of each model, providing insights into their performance and potential applications.

Convolutional Neural Networks (CNN): The CNN model achieved an accuracy of 93.12%, with precision, recall, and F1-score all at 93.00%. CNNs are particularly effective in capturing local dependencies and patterns in the text through convolutional layers. They excel in tasks where identifying specific phrases or local features is crucial. Additionally, CNNs are relatively faster to train compared to recurrent models like LSTM and GRU, owing to their ability to perform parallel computations. However, the slightly lower performance of CNNs in this task suggests that they might not capture long-term dependencies in the text as effectively as recurrent models. This limitation can be critical in sentiment analysis, where understanding the context over longer sequences is essential. Therefore, CNNs are best suited for scenarios requiring rapid inference with decent accuracy, such as real-time applications where processing speed is a priority.

Long Short-Term Memory (LSTM): The LSTM model demonstrated an accuracy of 95.33%, with a precision of 95.73%, recall of 94.89%, and an F1-score of 95.31%. LSTMs are designed to capture long-term dependencies in sequential data, making them highly effective for tasks like sentiment analysis that require contextual understanding over extended text. The architecture of LSTMs, which includes mechanisms for remembering long sequences, enables them to model the sequential nature of text effectively. However, LSTMs are computationally intensive and slower to train and infer compared to CNNs and transformer-based models. They also tend to require more memory, which can be a drawback in resource-constrained environments. LSTM models are ideal for applications where the understanding of the entire context is crucial, such as detailed sentiment analysis or processing long documents.

LSTM+GRU: The combined LSTM and GRU model achieved an accuracy of 95.61%, with a precision of 95.60%, recall of 95.63%, and an F1-score of 95.62%. This hybrid approach leverages the strengths of both LSTM and GRU units. GRUs are known for their ability to capture dependencies in sequences while being computationally more efficient than LSTMs. The combination of these two architectures allows the model to benefit from the robust sequence modeling capabilities of LSTMs and the computational efficiency of GRUs. This results in slightly better performance than standalone LSTM models. This combined model is suitable for tasks requiring both the detailed contextual understanding of LSTMs and the efficiency of GRUs, making it a balanced choice for many sentiment analysis applications.

Distil BERT: The Distil BERT model outperformed all other models, with an accuracy of 96.30%, a precision of 97.00%, recall of 96.00%, and an F1-score of 96.00%. Distil BERT, a distilled version of BERT, offers the benefits of transformer-based models, including the ability to capture intricate contextual relationships in text. It retains 97% of BERT's performance while being significantly faster and requiring fewer computational resources. This makes Distil BERT highly effective for sentiment analysis, achieving high accuracy and balanced precision and recall. The primary limitation of transformer-based models like Distil BERT is their computational intensity during training, though they are efficient during inference. Distil BERT is the best choice for applications requiring high accuracy and balanced performance metrics, such as sentiment analysis where both precision and recall are critical.

7.1 Model Selection for Specific Tasks

Each model has distinct strengths that make them suitable for different tasks within sentiment analysis:

- For applications requiring rapid inference, such as real-time sentiment analysis in social media monitoring, the CNN model is ideal due to its faster training and inference times. - When the task demands a deep understanding of long-term dependencies and context within the text, such as analyzing lengthy reviews or documents, LSTM models are more appropriate. - The LSTM+GRU model offers a balanced approach, providing both robust contextual understanding and computational efficiency. It is suitable for general-purpose sentiment analysis tasks where both performance and efficiency are important. - For tasks where high accuracy and balanced precision and recall are paramount, such as in critical sentiment analysis applications in customer feedback systems or market analysis, Distil BERT is the best choice.

In conclusion, the selection of a model should be based on the specific requirements of the task, including the need for accuracy, speed, and computational efficiency. Each model presents unique advantages, making them suitable for various aspects of sentiment analysis.

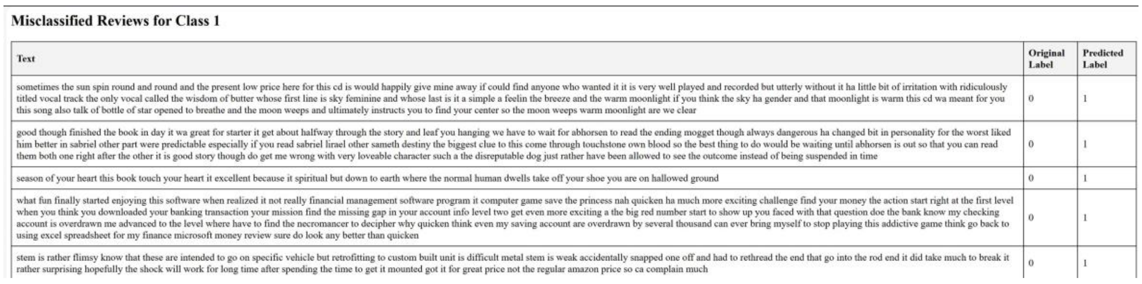
7.2 Discussion on Misclassified Reviews

To gain deeper insights into the performance of our best model, Distil BERT, we examined the reviews it incorrectly classified. Although machine learning models are often considered black boxes due to their complex and abstract nature, analyzing these misclassified instances can provide valuable understanding into potential patterns and shortcomings in the model’s learning process. We took the first 10 negative reviews i.e. label 0 that were classified as positive reviews i.e. label 1. Below, we tried to analyze what could be the reasons for these errors.

Analysis of Misclassified Reviews

Analysis of Text 1: The review contains a significant amount of sarcasm and irony, which can be challenging for the model to detect. Phrases like "I would happily give mine away" and "utterly without" indicate dissatisfaction. However, the model might have been misled by phrases such as "very well played and recorded" and "warm moonlight". The model likely focused on these seemingly positive aspects, leading to misclassification.

Analysis of Text 2: This review starts with positive remarks such as "finished the book in a day" and "great for a starter". Despite these initial positives, the review also contains critical comments.



Misclassified Reviews for Class 1

Text	Original Label	Predicted Label
sometimes the sun spin round and round and the present low price here for this cd is would happily give mine away if could find anyone who wanted it it is very well played and recorded but utterly without it ha little bit of irritation with ridiculously titled vocal track the only vocal called the wisdom of butter whose first line is sky feminine and whose last is it a simple a feelin the breeze and the warm moonlight if you think the sky ha gender and that moonlight is warm this cd wa meant for you this song also talk of bottle of star opened to breathe and the moon weeps and ultimately instructs you to find your center so the moon weeps warm moonlight are we clear	0	1
good though finished the book in day it wa great for starter it get about halfway through the story and leaf you hanging we have to wait for abhorsen to read the ending mogget though always dangerous ha changed bit in personality for the worst liked him better in sabriel other part were predictable especially if you read sabriel lirael other sameth destiny the biggest clue to this come through touchstone own blood so the best thing to do would be waiting until abhorsen is out so that you can read them both one right after the other it is good story though do get me wrong with very loveable character such a the disreputable dog just rather have been allowed to see the outcome instead of being suspended in time	0	1
season of your heart this book touch your heart it excellent because it spiritual but down to earth where the normal human dwells take off your shoe you are on hallowed ground	0	1
what fun finally started enjoying this software when realized it not really financial management software program it computer game save the princess nah quicken ha much more exciting challenge find your money the action start right at the first level when you think you downloaded your banking transaction your mission find the missing gap in your account info level two get even more exciting a the big red number start to show up you faced with that question doe the bank know my checking account is overdrawn me advanced to the level where have to find the necromancer to decipher why quicken think even my saving account are overdrawn by several thousand can ever bring myself to stop playing this addictive game think go back to using excel spreadsheet for my finance microsoft money review sure do look any better than quicken	0	1
stem is rather flimsy know that these are intended to go on specific vehicle but retrofitting to custom built unit is difficult metal stem is weak accidentally snapped one off and had to rethread the end that go into the rod end it did take much to break it rather surprising hopefully the shock will work for long time after spending the time to get it mounted got it for great price not the regular amazon price so ca complain much	0	1

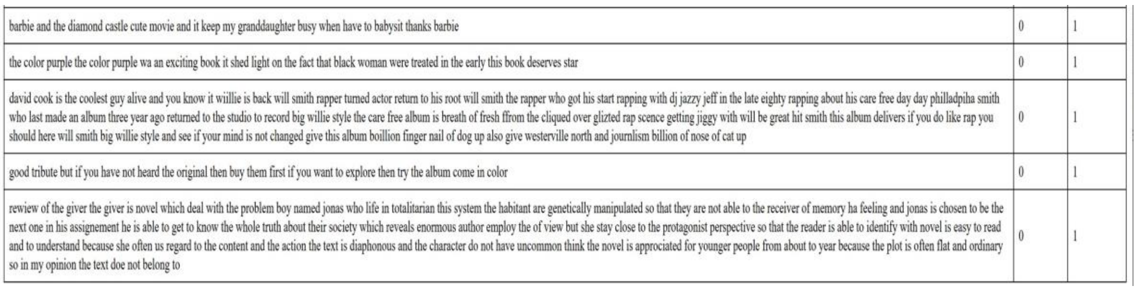
Figure 24: First five misclassified reviews for Class 1

like "predictable" and "changed bit in personality for the worst". The model may have been biased towards the positive opening statements, overshadowing the subsequent criticisms.

Analysis of Text 3: The review is a mix of positive and negative sentiments. While it contains praise like "touch your heart" and "excellent", the model might have missed the overall context or the more nuanced criticisms. This led to a misclassification as the model focused on the positive aspects.

Analysis of Text 4: This review uses a humorous and sarcastic tone to compare financial software to a game. Positive phrases like "what fun" and "addictive game" might have confused the model. The underlying criticism was likely obscured by these playful expressions, resulting in misclassification.

Analysis of Text 5: The review includes both positive ("great price") and negative sentiments ("rather flimsy", "weak"). The positive ending may have influenced the model's classification, despite the critical comments about the product's quality.



barbie and the diamond castle cute movie and it keep my granddaughter busy when have to babysit thanks barbie	0	1
the color purple the color purple wa an exciting book it shed light on the fact that black woman were treated in the early this book deserves star	0	1
david cook is the coolest guy alive and you know it wiillie is back will smith rapper turned actor return to his root will smith the rapper who got his start rapping with dj jazzy jeff in the late eighty rapping about his care free day day philladpiha smith who last made an album three year ago returned to the studio to record big willie style the care free album is breath of fresh ffrom the cliqued over glizted rap scence getting jiggy with will be great hit smith this album delivers if you do like rap you should here will smith big willie style and see if your mind is not changed give this album boillion finger nail of dog up also give westerville north and journlism billion of nose of cat up	0	1
good tribute but if you have not heard the original then buy them first if you want to explore then try the album come in color	0	1
rewiew of the giver the giver is novel which deal with the problem boy named jonas who life in totalitarian this system the habitant are genetically manipulated so that they are not able to the receiver of memory ha feeling and jonas is chosen to be the next one in his assignement he is able to get to know the whole truth about their society which reveals enormous author employ the of view but she stay close to the protagonist perspective so that the reader is able to identify with novel is easy to read and to understand because she often us regard to the content and the action the text is diaphonous and the character do not have uncommon think the novel is approciated for younger people from about to year because the plot is often flat and ordinary so in my opinion the text doe not belong to	0	1

Figure 25: 6th to 10th misclassified reviews for Class 1

Analysis of Text 6: The review appears mostly positive, but the implicit criticism might have been missed. The model could have been swayed by phrases like "cute movie" and "keep my granddaughter busy", leading to an incorrect classification.

Analysis of Text 7: This review is ambiguous and contains phrases like "exciting book" and "deserves star" that might have led the model to classify it as positive. The model may have missed the more subtle negative undertones related to the book’s content.

Analysis of Text 8: The review is incoherent and lacks clear sentiment. Positive mentions ("coolest guy alive", "breath of fresh air") likely influenced the model’s classification. The overall context was likely too confusing for the model to accurately interpret the sentiment.

Analysis of Text 9: The review starts positively but includes a recommendation that might imply the tribute is not as good as the original. The mixed sentiment and positive opening could have led the model to misclassify it as positive.

Analysis of Text 10: The review is critical but includes phrases like "novel is easy to read" and "to understand". These positive aspects might have led the model to classify it as positive, despite the overall negative sentiment.

By examining these misclassified reviews, we can identify areas where the Distil BERT model struggles, such as detecting sarcasm, handling mixed sentiments, and interpreting ambiguous or incoherent text. This analysis provides a foundation for future improvements to enhance the model’s robustness and accuracy.

8 Conclusion and Future Work

The journey of sentiment analysis on Amazon review texts has provided a thorough understanding of the capabilities and limitations of various machine learning models. Each modelCNN, LSTM, LSTM+GRU, and Distil BERTwas evaluated for its effectiveness in capturing the nuances of sentiment expressed in text, and each exhibited unique strengths suited to different applications.

The analysis of misclassified reviews using the Distil BERT model highlighted several areas where improvements are needed. Detecting sarcasm, handling mixed sentiments, and interpreting ambiguous or incoherent text remain challenging. Addressing these issues is critical for enhancing model robustness and accuracy.

Sarcasm detection is complex as it often relies on contextual clues and tone, which are difficult for models to capture. Sarcasm can invert the sentiment of a statement, making positive words convey a negative sentiment and vice versa. Future improvements could involve incorporating additional features or context-aware mechanisms to better understand the nuances of sarcasm. Techniques such as sentiment shift detection, contextual embeddings, and multimodal approaches (combining text with audio or visual cues) could be explored to enhance sarcasm detection capabilities.

Reviews with mixed sentiments present a unique challenge as they contain both positive and negative expressions. Such reviews can confuse models trained on distinctly positive or negative examples. Advanced models capable of segmenting and analyzing different parts of the text independently might offer better performance in such cases. Techniques like aspect-based sentiment analysis, where the sentiment towards specific aspects of a product is evaluated separately, can help in accurately capturing mixed sentiments. Incorporating hierarchical models that can parse and analyze sentences at multiple levels (word, sentence, paragraph) may also improve handling of mixed sentiments.

Ambiguity and incoherence in text can confuse models, leading to incorrect sentiment classification. Ambiguous statements often require a deep understanding of context or external knowledge, which current models may lack. Preprocessing techniques that enhance text clarity and coherence, along with models that can handle uncertainty, may improve performance. Approaches such as fuzzy logic, probabilistic models, and incorporating external knowledge bases (e.g., knowledge graphs) can be explored to better interpret ambiguous or incoherent text.

Our exploration underscores the importance of selecting the right model based on task-specific requirements. CNNs are ideal for speed-centric applications, LSTMs for in-depth contextual understanding, LSTM+GRU hybrids for balanced performance, and Distil BERT for highaccuracy needs. This journey has provided critical insights into the strengths and limitations of each model, guiding future advancements in the field of sentiment analysis for Amazon review texts.

The process of conducting sentiment analysis on Amazon review texts has been both challenging and enlightening. It has underscored the importance of thorough preprocessing, careful model selection, and refinement to address complex linguistic phenomena. As sentiment analysis technology continues to evolve, these insights will be invaluable for developing more sophisticated and accurate models capable of understanding human sentiment through text.

References

[1] Bing Liu. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers, 2012.

[2] Bo Pang and Lillian Lee. “Opinion Mining and Sentiment Analysis”. In: *Foundations and Trends in Information Retrieval*. Vol. 2. 2008, pp. 1–135.

[3] Minqing Hu and Bing Liu. “Mining and Summarizing Customer Reviews”. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2004, pp. 168–177.

[4] Julian McAuley and Jure Leskovec. “Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text”. In: *Proceedings of the 7th ACM Conference on Recommender Systems*. 2013, pp. 165–172.

[5] Ruining He and Julian McAuley. “Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering”. In: *Proceedings of the 25th International Conference on World Wide Web*. 2016, pp. 507–517.

[6] Yoon Kim. “Convolutional Neural Networks for Sentence Classification”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1746–1751.

[7] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9 (1997), pp. 1735–1780.

[8] Junyoung Chung et al. “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”. In: *NIPS 2014 Workshop on Deep Learning*. 2014.

[9] Victor Sanh et al. “Distil BERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *arXiv preprint arXiv:1910.01108* (2019).

[10] Rie Johnson and Tong Zhang. “Effective Use of Word Order for Text Categorization with Convolutional Neural Networks”. In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2015, pp. 103–112.

[11] Duyu Tang, Bing Qin, and Ting Liu. “Document Modeling with Gated Recurrent Neural Network for Sentiment Classification”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2015, pp. 1422–1432.

[12] Bin Zhou et al. “Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling”. In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2016, pp. 3485– 3495.

[13] Ashish Vaswani et al. “Attention is All you Need”. In: *arXiv preprint arXiv:1706.03762* (2017).

[14] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *arXiv preprint arXiv:1810.04805* (2019).

[15] Lei Zhang, Shuai Wang, and Bing Liu. “Deep Learning for Sentiment Analysis: A Survey”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8.4 (2018), e1253.

[16] Maite Taboada et al. “Lexicon-based methods for sentiment analysis”. In: *Computational linguistics* 37.2 (2011), pp. 267–307.

[17] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. “Thumbs up?: sentiment classification using machine learning techniques”. In: *Proceedings of the ACL-02 conference*

on Empirical methods in natural language processing-Volume 10. Association for Computational Linguistics. 2002, pp. 79–86.

[18] Fabrizio Sebastiani. “Machine learning in automated text categorization”. In: *ACM computing surveys (CSUR)* 34.1 (2002), pp. 1–47.

[19] Jacob Devlin et al. “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[20] Victor Sanh et al. “Distil BERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *arXiv preprint arXiv:1910.01108* (2019).

[21] Yoon Kim. “Convolutional neural networks for sentence classification”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1746–1751.

[22] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[23] Matthias Bittlingmayer. *Amazon Reviews for Sentiment Analysis*. Accessed: 2024-03-24. 2017. URL: https://www.kaggle.com/datasets/bittlingmayer/ amazonreviews.

[24] Tomas Mikolov et al. “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781* (2013).

[25] Nitish Srivastava et al. “Dropout: A simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[26] Mike Schuster and Kuldip K Paliwal. “Bidirectional recurrent neural networks”. In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681.

[27] Min Lin, Qiang Chen, and Shuicheng Yan. “Network in network”. In: *arXiv preprint arXiv:1312.4400* (2013).

[28] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

[29] Geoffrey E Hinton et al. “Improving neural networks by preventing co-adaptation of feature detectors”. In: *arXiv preprint arXiv:1207.0580* (2012).

[30] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[31] Bing Liu. *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers, 2012. [32] Tom Young et al. “Recent trends in deep learning based natural language processing”. In: *ieee Computational intelligenCe magazine* 13.3 (2018), pp. 55–75.

[33] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O’Reilly Media, Inc.", 2009.

[34] Matthew Honnibal and Ines Montani. “spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing”. In: *To appear* 7 (2017), pp. 411–420.

[35] Tomas Mikolov et al. “Recurrent neural network based language model”. In: *Interspeech*. Vol. 2. 2010, pp. 1045–1048.

[36] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[37] Junyoung Chung et al. “Empirical evaluation of gated recurrent neural networks on sequence modeling”. In: *arXiv preprint arXiv:1412.3555* (2014).

[38] Klaus Greff et al. “LSTM: A search space odyssey”. In: *IEEE transactions on neural networks and learning systems* 28.10 (2017), pp. 2222–2232.

[39] Junyoung Chung et al. “A recurrent latent variable model for sequential data”. In: *Advances in neural information processing systems*. 2015, pp. 2980–2988.