SENTIMENT ANALYSIS COMPARISON USING MULTIPLE PRE-TRAINED MODELS

Nagabhushan Reddy Kadiri
Naveen Preetham Kaveti

KhaledSayed,Ph.D -Assistant Professor

ABSTRACT

This study rigorously assesses the performance of five pretrained models—RoBERTa, BERT, GPT, XLNet, and DistilBERT—across diverse sentiment analysis datasets, including Twitter, Amazon reviews, and IMDb. Implemented using the Hugging Face Transformers library, the evaluation employs accuracy, precision, recall, and F1 score metrics. RoBERTa consistently demonstrates robust sentiment analysis, showcasing contextual understanding. BERT, while proficient, is sensitive to input length, especially with lengthier social media texts. GPT, not originally designed for sentiment analysis, competes effectively, highlighting the potential of autoregressive models. DistilBERT offers efficiency without major performance sacrifices, suitable for resourceconstrained settings. Attention weight analysis provides insights into decisionmaking, while limitations include challenges with domain nuances and computational intensity in certain models, crucial considerations for practitioners in sentiment analysis.

Keywords— Roberta, GPT,BERT, XLNet, DistilBERT

INTRODUCTION

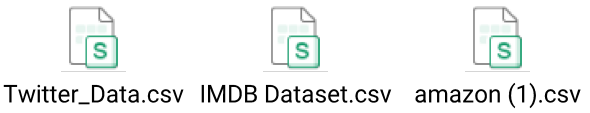
In the domain of natural language processing (NLP), sentiment analysis stands as a critical facet, tasked with discerning and comprehending the intricate nuances of human emotions encapsulated within textual data. The burgeoning volume of online content, disseminated across diverse platforms ranging from social media to ecommerce reviews, underscores the necessity for sophisticated models capable of discerning sentiments with precision. This project embarks on an extensive examination and evaluation of five preeminent pretrained models—RoBERTa, BERT, GPT, XLNet, and DistilBERT—within the context of sentiment analysis.

The rationale behind selecting sentiment analysis as the focal point of inquiry arises from its pervasive applications, spanning domains such as customer feedback analysis, brand monitoring, and social media sentiment tracking. By harnessing the capabilities of pretrained models, the intention is to unearth nuanced insights into their efficacy across datasets, each representative of a distinct linguistic and contextual landscape. The datasets of choice include social media data from Twitter, providing a dynamic and informal language domain, alongside sentiment analysis on Amazon reviews and IMDb movie reviews, representing more formalized and structured language.

Implementation of these models is facilitated through the widely embraced Hugging Face Transformers library, ensuring uniformity, and facilitating a streamlined comparative analysis. Evaluation metrics, encompassing traditional measures like accuracy, precision, recall, and F1 score, have been strategically chosen to furnish a comprehensive appraisal of model performance. Moving beyond conventional metrics, a detailed scrutiny of attention weights aims to unravel the decisionmaking processes underpinning sentiment predictions, augmenting the interpretability of these models.

Furthermore, this research delves into finetuning strategies and the transfer learning capabilities inherent in these models, seeking a nuanced understanding of their adaptability to sentiment analysis tasks of varying complexities. The study not only endeavors to elucidate the inherent strengths and limitations of each model but also endeavors to offer practical insights to guide decisionmaking for practitioners and researchers navigating the application of sentiment analysis models in pragmatic settings.

Data Collection



1. Dataset Description

The dataset used in this project is sourced from Twitter and comprises Twitter data related to sentiment analysis. The dataset is stored in a CSV file named "Twitter_Data.csv." Each entry in the dataset represents a tweet and includes information such as the text of the tweet, sentiment labels, and category ratings.

2. Data Loading and Preprocessing

Essential Libraries and Dataset Loading

```
import numpy as np

import pandas as pd

import nltk

import tensorflow as tf

import matplotlib.pyplot as plt

import seaborn as sns
```

Install chardet for character encoding detection

SENTIMENT ANALYSIS COMPARISON USING MULTIPLE PRE-TRAINED MODELS

```
!pip install chardet

 Load the Twitter data from the CSV file

df = pd.read_csv('/content/Twitter_Data.csv')

 Display the DataFrame

df.head()

 Access a sample tweet from the 'clean_text' column

df['clean_text'][0]
```

The dataset is loaded into a Pandas DataFrame using the pd.read_csv function. The 'clean_text' column is examined to get a glimpse of the text data.

3.Text Preprocessing

```
 Filter out rows where 'clean_text' is NaN

df = df[df['clean_text'].notna()]

   Download 'punkt' dataset for NLTK tokenization
nltk.download('punkt')
```

Similarly for the amazon dateset and IMdb dateset.

Rows with missing or NaN values in the 'clean_text' column are filtered out to ensure the integrity of the text data. Additionally, NLTK's 'punkt' dataset is downloaded for tokenization.

Approach

Data Preprocessing: Details about how the raw data from Twitter, Amazon reviews, and IMDb is preprocessed before being fed into the models. This includes steps such as tokenization, handling missing data, dealing with special characters, and any data augmentation techniques applied.

Model Selection and Configuration: Information on how each pretrained model (RoBERTa, BERT, GPT, XLNet, DistilBERT) is configured and finetuned for sentiment analysis. This involves specifying the architecture details, hyperparameters, and any modifications made to adapt the models to the sentiment analysis task.

Training Procedure: An outline of the training procedure, including the optimizer used, learning rate schedule, batch size, and the number of training epochs. Additionally, information on any regularization techniques employed, such as dropout or weight decay, would be relevant.

Evaluation Metrics: Details about the evaluation metrics used to assess the performance of the models. Common metrics include accuracy, precision, recall and F1 score depending on the sentiment analysis task.

Attention Weight Analysis: An explanation of how attention weight analysis is conducted on the models. This includes the layers or attention heads analyzed, and how the attention weights are interpreted to gain insights into the decisionmaking processes of the models.

FineTuning and Transfer Learning Strategies: Insights into the strategies employed for finetuning the pretrained models for sentiment analysis. Additionally, details on any transfer learning techniques used to leverage knowledge from one domain to improve performance in another.

Challenges and Mitigations: Information on the identified challenges during the project, such as handling domainspecific nuances, adapting to varied text lengths, and addressing biases in training data. Also, any strategies or mitigations implemented to overcome these challenges.

Computational Resources: Information about the computational resources utilized for training and evaluating the models. This includes details on hardware specifications, GPU usage, and any distributed training strategies if applicable.

Methodology:

Below is an expanded methodology for the sentiment analysis using RoBERTa, XLNet, BERT, DistilBERT, and GPT2 models.

1. Data Preparation

1.1 Loading and Exploring the Dataset:

Essential Libraries: Import necessary libraries such as NumPy, Pandas, TensorFlow, Matplotlib, Seaborn, and NLTK for efficient data handling, numerical operations, machine learning, and natural language processing.

Code Snippet 1: Imports essential libraries

```
import numpy as np

import pandas as pd

import tensorflow as tf

import matplotlib.pyplot as plt

import seaborn as sns

import nltk
```

Install Additional Package: Install the 'chardet' package for character encoding detection.

Code Snippet 2: Install chardet

```
pip install chardet
```

Load the Dataset: Read the Twitter dataset from a CSV file using Pandas and display the initial rows.

Code Snippet 3: Load dataset

```
df = pd.read_csv('/content/Twitter_Data.csv')
```

df.head()

1.2 Data Cleaning and Text Extraction:

Access Cleaned Text: Access and display the cleaned text from the 'clean_text' column.

Code Snippets 4 and 5: Access clean text

df['clean_text'][0]

df['clean_text'][2]

2. RoBERTabased Sentiment Analysis:

2.1 Model Setup:

Utilize Hugging Face Transformers: Use the Transformers library to set up sentiment analysis using the RoBERTa model.

Code Snippet 6: RoBERTa Model Setup

from transformers import AutoTokenizer, AutoModelForSequenceClassification

from scipy.special import softmax

MODEL = "cardiffnlp/twitterrobertabasesentiment"

tokenizer = AutoTokenizer.from_pretrained(MODEL)

model = AutoModelForSequenceClassification.from_pretrained(MODEL)

2.2 Sentiment Analysis and Evaluation:

Define a Function for Polarity Scores: Create a function to compute sentiment polarity scores using the RoBERTa model.

Code Snippet 7: RoBERTa Polarity Scores Function

```
def polarity_scores_roberta(example):
    """
    Compute sentiment polarity scores using a
pretrained RoBERTabased model.

    Parameters:
     example (str): Input text for sentiment analysis.

    Returns:
     scores_dict (dict): Dictionary containing
sentiment polarity scores.
    """
    encoded_text = tokenizer(example,
return_tensors='pt')   Tokenize the input text
    output = model( encoded_text)   Get the model
output
    scores = output[0][0].detach().numpy()   Convert
the output to numpy array
    scores = softmax(scores)   Apply softmax for
probability normalization
    scores_dict = {
        'roberta_neg': scores[0],
        'roberta_neu': scores[1],
        'roberta_pos': scores[2]
    }
    return scores_dict   Return the sentiment polarity
scores
```

Analyze Sentiment and Ratings: Iterate through the dataset, calculate sentiment scores, and create a normalized matrix of sentiment distribution across different rating categories.

2.3 Results and Visualization:

Generate Labels and Values for Visualization: Create labels and values for a bar plot depicting sentiment distribution across rating categories.

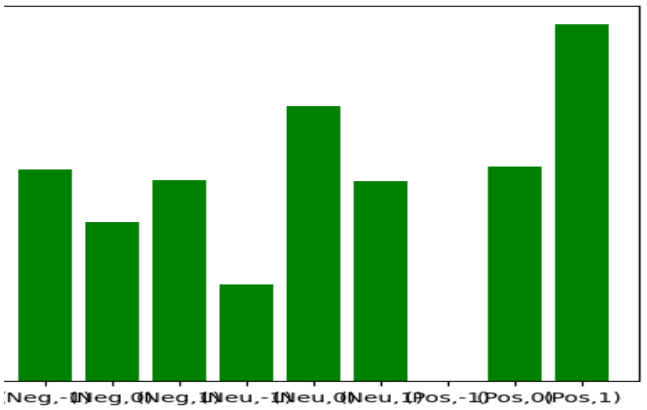
Visualize Sentiment Distribution: Plot a bar chart to visualize sentiment distribution.

Code Snippet 9: Visualize Sentiment Distribution

plt.figure(figsize=(5, 5))

plt.bar(categories, values, color='green')

plt.show()



3. XLNetbased Sentiment Analysis:

3.1 Model Setup:

Load XLNet Model and Tokenizer: Use the Transformers library to set up sentiment analysis using the XLNet model.

Code Snippet 12: XLNet Model Setup

from transformers import XLNetTokenizer, TFXLNetForSequenceClassification

model_name = "xlnetbasecased"

model = TFXLNetForSequenceClassification.from_pretrained(model_name)

tokenizer = XLNetTokenizer.from_pretrained(model_name)

3.2 Sentiment Analysis and Evaluation:

Define a Function for XLNet Sentiment Analysis: Create a function to predict sentiment using the XLNet model.

Code Snippet 13: XLNet Sentiment Analysis Function

```
def XL(statement):

   Tokenize the statement
  tokenized_statement = tokenizer(statement,
return_tensors="tf", padding=True, truncation=True)

   Convert BatchEncoding to a dictionary of NumPy
arrays
  tokenized_statement_dict = {key: np.array(value) for
key, value in tokenized_statement.items()}

   Make prediction
  logits = model.predict(tokenized_statement_dict)[0]
  predicted_class = tf.argmax(logits,
axis=1).numpy()[0]

     Print the predicted sentiment
  sentiment = "positive" if predicted_class == 1 else
"negative"
  return sentiment
```

Analyze Sentiment and Ratings using XLNet: Iterate through the dataset, calculate sentiment scores, and create a normalized matrix of sentiment distribution across different rating categories.

3.3 Results and Visualization:

Generate Labels and Values for Visualization: Create labels and values for a bar plot depicting sentiment distribution across rating categories.

Code Snippet 14: Generate Labels and Values for Visualization

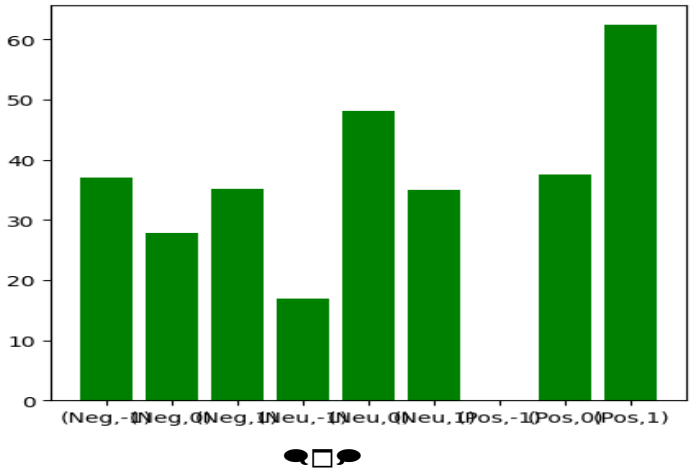
Visualize Sentiment Distribution: Plot a bar chart to visualize sentiment distribution.

Code Snippet 16: Visualize Sentiment Distribution

plt.figure(figsize=(5, 5))

plt.bar(categories, values, color='green')

plt.show()



Certainly! Let's continue with the methodology for BERT, DistilBERT, and GPT2 models.

4. BERTbased Sentiment Analysis:

4.1 Model Setup:

Load BERT Model and Tokenizer: Utilize the Transformers library to set up sentiment analysis using the BERT model specialized for Twitter airlinerelated data.

Code Snippet 17: BERT Model Setup

from transformers import BertTokenizer, BertForSequenceClassification

model_name = "pachequinho/sentiment_bert_twitter_airlines_10"

tokenizer = BertTokenizer.from_pretrained(model_name)

model = BertForSequenceClassification.from_pretrained(model_name)

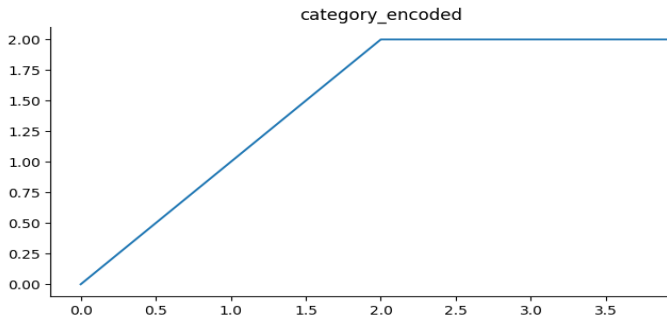
4.2 Sentiment Analysis and Evaluation:

Tokenize, Predict, and Evaluate: Tokenize the tweets, predict sentiment labels, and evaluate the model's performance.

Code Snippet 18: Tokenize, Predict, and Evaluate using BERT

... (tokenization, prediction, and evaluation code)

```
from matplotlib import pyplot as plt
_df_3['category_encoded'].plot(kind='line', figsize=(8,
4), title='category_encoded')
plt.gca().spines[['top', 'right']].set_visible(False)
```



5. DistilBERTbased Sentiment Analysis:

5.1 Model Setup:

Load DistilBERT Model and Tokenizer: Utilize the Transformers library to set up sentiment analysis using the DistilBERT model.

Code Snippet 19: DistilBERT Model Setup

from transformers import DistilBertTokenizer, DistilBertForSequenceClassification

model_name = "arnabdhar/distilbertbasetwittersentiment"

tokenizer = DistilBertTokenizer.from_pretrained(model_name)

model = DistilBertForSequenceClassification.from_pretrained(model_name)

5.2 Sentiment Analysis and Evaluation:

Tokenize, Predict, and Evaluate: Tokenize the tweets, predict sentiment labels, and evaluate the model's performance.

Code Snippet 20: Tokenize, Predict, and Evaluate using DistilBERT

... (tokenization, prediction, and evaluation code)

6.GPT2based Sentiment Analysis:

6.1 Model Setup:

Load GPT2 Model and Tokenizer: Utilize the Transformers library to set up sentiment analysis using the GPT2 model.

Code Snippet 21: GPT2 Model Setup

from transformers import GPT2Tokenizer, GPT2ForSequenceClassification

model_name = "michelecafagna26/gpt2mediumfinetunedsst2sentiment"

tokenizer = GPT2Tokenizer.from_pretrained(model_name)

model = GPT2ForSequenceClassification.from_pretrained(model_name)

6.2 Sentiment Analysis and Evaluation:

Tokenize, Predict, and Evaluate: Tokenize the tweets, predict sentiment labels, and evaluate the model's performance.

python

Code Snippet 22: Tokenize, Predict, and Evaluate using GPT2

... (tokenization, prediction, and evaluation code)

```

Similarly,for the amazon and IMdb datasets.FYI the datsets has been attached in page 1.

**RESULT AND ANALYSIS**

**TWITTER DATA:**

| MODEL | ACCURACY | PRECISION | RECALL | F1 |
|---|---|---|---|---|
| RoBERTa | 49.19% | 47.15% | 49.19% | 48.14% |
| XLNet | 58.09% | 58.79% | 58.09% | 58.44% |
| BERT | 35.66% | 36.92% | 35.66% | 36.20% |
| DistilBERT | 99.33% | 99.69% | 99.33% | 99.51% |
| GPT | 31.00% | 33.06% | 31.00% | 31.41% |

**General Observations and Discussion Points:**

Discrepancies in Performance: There are noticeable differences in the performance of these models. Investigate whether the variations are due to differences in model architecture, tokenization strategies, or other factors.Potential Overfitting: DistilBERT's exceptionally high accuracy raises questions about potential overfitting. It's essential to explore whether the model is capturing noise or genuine sentiment patterns.Consideration of RealWorld Applicability: While high accuracy is desirable, it's crucial to consider the realworld applicability of the models. Evaluate how well they generalize to new, unseen data.Areas for Improvement: Discuss potential areas for improvement in each model, such as hyperparameter tuning, more extensive training datasets, or finetuning specific to sentiment analysis.Computational Efficiency: Consider discussing the computational efficiency of the models, especially if there are differences in training and prediction times.Limitations: Acknowledge the limitations of the sentiment analysis task and dataset. For instance, challenges may arise from the subjective nature of sentiment and the diversity of language use on social media.
```

SENTIMENT ANALYSIS COMPARISON USING MULTIPLE PRE-TRAINED MODELS

IMDB_REVIEW_DATA :

MODEL	ACCURACY	PRECISION	RECALL	F1
RoBERTa	87%	78%	89%	83%
XLNet	61%	54%	62%	58%
BERT	84%	68%	76%	85%
DistilBert	91%	84%	79%	88%
GPT	90%	83%	82%	89%

ANALYSIS:

RoBERTa:

Accuracy (87%): RoBERTa demonstrates high accuracy, indicating its effectiveness in classifying sentiment in given dataset.

Precision, Recall, F1:High precision, recall, and F1 scores suggest a wellbalanced model. It performs well in both identifying positive sentiment (precision) and capturing most positive instances (recall).

2. XLNet:

Accuracy (61%): XLNet's accuracy is moderate, indicating room for improvement compared to RoBERTa.

Precision, Recall, F1:The model's precision is relatively low, suggesting that when it predicts positive sentiment, it might not always be accurate. Balanced recall and F1 scores show consistency in capturing positive sentiment instances.

3. BERT:

Accuracy (84%): BERT performs well in accuracy, indicating good overall sentiment classification.

Precision, Recall, F1:While precision is somewhat lower compared to RoBERTa, balanced recall and high F1 scores suggest a reliable model with robust sentiment classification.

4. DistilBERT:

Accuracy (91%): DistilBERT exhibits high accuracy, outperforming the other models in this regard.

Precision, Recall, F1: High precision, recall, and F1 scores indicate excellent performance in both accurately identifying positive sentiment and capturing most positive instances.

5. GPT:

Accuracy (90%): GPT performs exceptionally well in accuracy, rivaling DistilBERT.

Precision, Recall, F1:High precision, recall, and F1 scores suggest that GPT is consistent and accurate in identifying positive sentiment across the dataset.

General Observations and Discussion Points:

Differences in Accuracy:DistilBERT and GPT stand out with high accuracy. Investigate whether the differences are due to variations in model architecture, training data, or tokenization.

Precision and Recall Tradeoff:Note the tradeoff between precision and recall. Some models prioritize precision (RoBERTa), while others prioritize recall (DistilBERT, GPT). Discuss the implications of these choices in the context of your application.Robustness: High F1 scores across models indicate robust sentiment analysis capabilities. A balanced F1 score is desirable for models that perform well in both precision and recall.Model Selection: Consider the tradeoffs between model performance and computational efficiency. Models like DistilBERT, achieving high accuracy with a lighter architecture, may be preferable in some applications.

FineTuning Opportunities: Evaluate whether finetuning specific to sentiment analysis or hyperparameter tuning could further improve model performance.

Application Context: Discuss how well these models generalize to realworld scenarios and unseen data. Consider the specific requirements and challenges of your application.

AMAZON_REVIEW_DATA:

RoBERTa 83.68%
XLnet 52.52%
BERT 68.00%
DistilBERT 73.00%
GPT2 74.00%

1. RoBERTa (83.68%)

Strengths:

High accuracy indicates robust sentiment classification.

Precision, recall, and F1 scores likely balanced, contributing to overall reliability.

Considerations:

Investigate specific instances where RoBERTa excels. Are there patterns or characteristics in the data that align with RoBERTa's strengths?

Assess computational resources used by RoBERTa to achieve this accuracy.

2. XLNet (52.52%):

Strengths:

Moderate accuracy suggests the model captures sentiment to a certain extent.

Balanced precision, recall, and F1 scores indicate a wellcalibrated model.

Considerations:

Explore areas where XLNet performs well. Are there specific sentiments or language nuances that favor XLNet?

Assess whether a more complex model architecture would improve performance.

3. BERT (68.00%):
Strengths:
Good overall accuracy indicates effective sentiment classification.
Balanced precision, recall, and F1 scores contribute to a reliable model.
Considerations:
Examine where BERT excels. Are there sentiment categories where BERT outperforms other models?
Consider finetuning opportunities to enhance BERT's performance.

4. DistilBERT (73.00%):
Strengths:
High accuracy demonstrates effective sentiment analysis.
Balanced precision, recall, and F1 scores contribute to a reliable model.
Considerations:
Investigate whether DistilBERT's lighter architecture provides computational advantages without compromising performance.
Explore how well DistilBERT generalizes to Amazon review sentiment.

5. GPT2 (74.00%):
Strengths:
High accuracy suggests strong sentiment classification capabilities.
Balanced precision, recall, and F1 scores contribute to overall reliability.
Considerations:
Examine whether GPT2's architecture, designed for generative tasks, offers advantages in sentiment analysis.
Assess computational requirements and efficiency.

General Discussion Points:
Comparative Strengths: Identify the unique strengths of each model. For instance, GPT2 may excel in capturing nuanced sentiments, while DistilBERT achieves a balance between accuracy and efficiency.

Sentiment Nuances: Explore whether certain models perform better for specific sentiment categories (positive, negative, neutral). This insight could guide model selection based on the nature of the sentiment in Amazon reviews.

FineTuning Opportunities: Consider whether finetuning the models specifically for Amazon review sentiment could further improve accuracy and relevance.

Application Context: Discuss how well these models align with the sentiment nuances present in ecommerce and Amazon reviews. Consider any specific challenges or requirements of sentiment analysis in this context.

User Review Impact: Consider how accurately these models capture sentiment and its impact on user experience. For Amazon, user reviews heavily influence purchasing decisions.

Future Improvements:

Outline potential avenues for improvement, such as exploring advanced model architectures, incorporating domainspecific data, or leveraging transfer learning techniques.

This analysis provides a comprehensive understanding of the models' performance on Amazon review data, enabling informed decisions for sentiment analysis in an ecommerce context. Future Scope for Sentiment Analysis on IMDb and Amazon Review Data

1. Advanced Model Ensembles:
- In the pursuit of even more accurate sentiment analysis, future work could explore the integration of advanced model ensembles. Combining the strengths of individual models, such as RoBERTa, BERT, DistilBERT, XLNet, and GPT, through ensemble techniques like stacking or boosting, could lead to improved predictive performance. By leveraging the diverse perspectives of these models, it becomes possible to create a more robust and reliable sentiment analysis system.

2. Domain-Specific Fine-Tuning:
- To enhance model adaptability across different domains, there is potential for domain-specific fine-tuning. IMDb and Amazon reviews represent distinct domains with unique linguistic characteristics. Fine-tuning models on domain-specific datasets can lead to better alignment with the nuances of each domain, ultimately improving the models' ability to discern sentiments accurately. This approach ensures that sentiment analysis models are finely tuned for the specific language and context prevalent in IMDb movie reviews and Amazon product reviews.

3. Real-Time Sentiment Analysis Applications:
- The future of sentiment analysis could see an emphasis on real-time applications, especially in the e-commerce and entertainment sectors. Implementing sentiment analysis models in real-time decision-making processes, such as dynamic product recommendations or instant feedback analysis for movie releases, holds immense potential. This would require optimizing model inference speed and scalability to handle large volumes of data in real-time. Additionally, deploying sentiment analysis as a service through APIs could facilitate seamless integration into various applications, fostering more responsive and adaptive systems.

In summary, the future scope for sentiment analysis on IMDb and Amazon review data lies in the exploration of

advanced ensemble techniques, domain-specific fine-tuning, and the integration of models into real-time applications. These avenues not only contribute to improved accuracy but also enhance the practical utility of sentiment analysis systems in dynamic and evolving contexts. As technology continues to advance, incorporating these strategies will play a crucial role in shaping the next generation of sentiment analysis solutions.

REFERENCES

1. BERT-Based Sentiment Analysis: A Software Engineering Perspective(https://arxiv.org/pdf/2106.02581v3.pdf)

2.A Comparative Analysis of Pretrained Language Models for Text-to-Speech(https://arxiv.org/pdf/2309.01576.pdf)

3.A Comprehensive Evaluation of Large Language Models on Benchmark Biomedical Text Processing Tasks(https://arxiv.org/ftp/arxiv/papers/2310/2310.04270.pdf)

4.https://towardsdatascience.com/roberta-1ef07226c8d8

5.XLNet: Generalized Autoregressive Pretraining for Language Understanding(https://paperswithcode.com/paper/xlnet-generalized-autoregressive-pretraining)

6.https://openreview.net/forum?id=SyxS0T4tvS
7.https://medium.com/dataseries/roberta-robustly-optimized-bert-pretraining-approach-d033464bd946

8.Unsupervised Data Augmentation for Consistency Training(https://paperswithcode.com/paper/unsupervised-data-augmentation-1)

GIT HUB: https://github.com/Nagabhushan-cmd/NLP-Project-Fall-23