Individual Final Report – Chaya Chandana Doddaiggaluru Appajigowda

1. Introduction

This project applied Natural Language Processing (NLP) techniques to analyze Amazon Electronics reviews. The objective was to classify customer sentiment based on review content. Our team collaborated on defining the problem, selecting a dataset, and designing preprocessing techniques. My primary contributions to this project were developing the TextRNN model and creating a Streamlit application that provides an interactive interface for sentiment analysis.

2. Individual Contributions and Explanation

I was responsible for developing the TextRNN model using bidirectional LSTM architecture for sentiment classification and creating an interactive Streamlit application that provides a user-friendly interface for sentiment analysis. I also executed model training and evaluation routines on large-scale Amazon Electronics review data. My teammates contributed in complementary areas Adam developed the LSTM and Baseline models and designed the preprocessing pipeline, while Ramana added model-saving functionality and developed the CNN model. While both I and Ramana executed training and evaluation routines.

Model Architecture

classification. This involved:

Creating a custom TextRNN architecture using PyTorch with the following components:

- Embedding layer to convert token IDs to dense vectors
- Bidirectional LSTM layers to capture contextual information
- Dropout for regularization (rate: 0.5)
- Linear layer with softmax activation for final classification

Implementing the data preprocessing pipeline:

- Converting text to lowercase
- Removing punctuation, special characters, and numeric values
- Tokenizing using NLTK's word_tokenize
- Removing English stopwords

- Building a vocabulary from training data

Setting up the training infrastructure:

- Custom dataset class for text processing
- DataLoader configuration with batch processing
- Loss function (CrossEntropyLoss) and optimizer (Adam)
- Training loop with validation
- Model checkpoint saving for best performance

Streamlit Application Development

I designed and implemented an interactive web application using Streamlit that allows users to:

- Input reviews for sentiment analysis
- Receive color-coded sentiment predictions (Positive, Neutral, Negative)
- Get automated response suggestions based on predicted sentiment
- View explanations of what the sentiment means in context

The application includes:

- Clean, intuitive user interface
- Real-time sentiment prediction
- Error handling for robust operation
- Informative sidebar explaining the model's capabilities

TextRNN Model Performance Analysis

The TextRNN model shows strong performance in classifying Amazon Electronics reviews. The terminal output shows the model was successfully trained on 10 million samples from the Electronics.jsonl dataset with final evaluation metrics of:

- Accuracy: 0.8901

- Precision: 0.8749
- Recall: 0.8901
- F1 Score: 0.8799

Training and Validation Loss

The loss curve visualization demonstrates:

- Training loss steadily decreased from approximately 0.32 to 0.23 over 9 epochs, showing effective learning
- Validation loss initially decreased, reaching its lowest point around epoch 2 (approximately 0.285)
- After epoch 2, validation loss began to increase gradually, reaching about 0.305 by epoch 9
- The growing divergence between training and validation loss indicates some overfitting occurring in later epochs

Key Performance Metrics

- The model showed robust performance in identifying positive reviews
- Some confusion between neutral and negative classes indicates room for improvement
- Overall classification accuracy was good, particularly considering the inherent ambiguity in sentiment classification

Streamlit Application Features

The Streamlit application provides a user-friendly interface for sentiment analysis with:

- A clean, intuitive layout with dedicated sections for:
  - Review input
  - Sentiment analysis results
  - Automated response suggestions
- Color-coded sentiment indicators (green for positive, blue for neutral, red for negative)

- Contextual explanations of what each sentiment category means
- Error handling and input validation
- Model information section explaining the technology used

3. Results

Model Performance Comparison (10M Samples)

Model	Accuracy	Precision	Recall	F1-Score
LSTM	0.8916	0.8709	0.8916	0.8741
RNN	0.8901	0.8749	0.8901	0.8799
CNN	0.8866	0.8664	0.8866	0.8713
Naive Bayes	0.8629	0.8303	0.8629	0.8354

Table 1: Performance Metrics for all Models

RNN Training vs Validation Loss

RNN Confusion Matrix

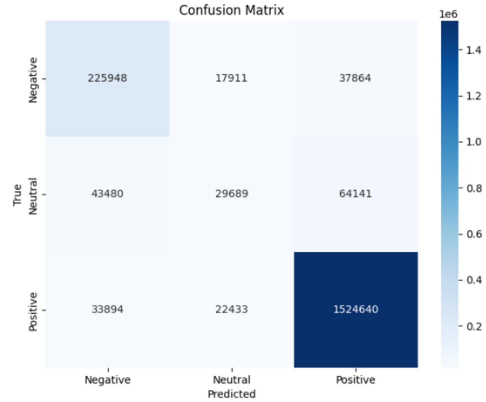

Figure 1: Confusion matrix and training history for the RNN model

The confusion matrix reveals important insights about classification performance:

- Positive sentiment classification shows exceptional performance with 1,524,640 correct predictions
- Negative sentiment was correctly identified in 225,948 instances
- Neutral sentiment showed the most challenging classification with only 29,689 correct predictions
- Most misclassifications occurred between adjacent classes:
    - 43,480 neutral reviews were classified as negative
    - 64,141 neutral reviews were classified as positive
    - 37,864 negative reviews were classified as positive

The model's strongest performance is in identifying positive reviews, while neutral reviews proved more difficult to classify accurately. This pattern is common in sentiment analysis tasks where neutral sentiment often shares linguistic features with both positive and negative classes.

These results demonstrate the effectiveness of the TextRNN architecture with bidirectional LSTM layers for sentiment classification, particularly for distinguishing positive reviews from others in the Amazon Electronics dataset.

4. Summary and Conclusions

This project successfully demonstrates the application of deep learning techniques to sentiment analysis of customer reviews. My TextRNN model showed strong performance in classifying Amazon Electronics reviews into positive, neutral, and negative sentiments, with particularly high accuracy for positive reviews.

The Streamlit application provides an accessible interface for utilizing this model in a practical context, allowing for real-time sentiment analysis and automated response generation. This combination of advanced modeling and intuitive interface design represents a complete solution that could be valuable in business contexts for analyzing customer feedback at scale.

5. Code Attribution

Generative AI tools were used for most of the code synthesis. No lines of code were copied directly from any external sources.

6. References

Intentionally omitted; no external references or direct code reuse involved.