CENG 442: Natural Language Processing
Assignment 2

Faruk Kaplan
December 23, 2024

Abstract

This sentiment analysis project uses a bidirectional LSTM model with extensive text preprocessing to ensure clean input. The model achieved 68.92% accuracy, excelling in classifying positive and negative sentiments. Pre-trained embeddings and dropout layers enhanced performance and prevented overfitting, ensuring strong generalization to unseen data.

Sentiment Analysis Report

The implemented code is accessible by opening the sentiment_analysis.ipynb file

1. Preprocessing Decisions

There are 8 different preprocessing steps that have been used. These are:

- Contraction expansion
- E-mail removal
- HTML tag removal
- URL removal
- Special character removal
- Accented character removal
- Text normalization
- Spelling correction

Expanding Contractions

Handling contractions was a key preprocessing step to ensure text uniformity in this sentiment analysis project. The contractions library was chosen for its efficiency in expanding shortened forms (e.g., "can't" to "cannot"), providing a standardized input. A custom dictionary was avoided due to the dataset's scale, making a ready-made solution more practical.

Expanding contractions reduces ambiguity, improving the model's ability to recognize patterns and enhancing overall performance in sentiment classification.

E-Mail Removal

To eliminate noise from the text data, email addresses and user handles were removed using custom regex patterns. This targeted approach efficiently captures and deletes emails and handles (e.g., "@username"), commonly found in social media and forum data.

By combining these patterns, irrelevant information is filtered out in one step, allowing the model to focus solely on meaningful text, enhancing sentiment analysis accuracy.

HTML Tag Removal

HTML tags were removed using a custom regex function to clean the text and eliminate artifacts from web-scraped or user-generated data. The regex targets text within angle brackets (<>), effectively stripping HTML elements.

BeautifulSoup library was implemented previously but it was inefficient, so custom regex was used.

This step enhances readability and ensures the model focuses on content rather than formatting, improving sentiment analysis accuracy.

URL Removal

URLs were removed using a custom regex to reduce noise and retain meaningful text. This is crucial for social media, blogs, and forums where links can obscure content.

The regex detects URLs starting with http://, https://, or www., ensuring clean text for better sentiment analysis. By eliminating links, the model can focus on the core message, improving accuracy.

Special Characters Removal

Special characters were removed using a custom regex to standardize the text and eliminate non-alphabetic symbols. This prevents irrelevant symbols from interfering with sentiment analysis.

The pattern preserves accented characters and whitespace, removing punctuation, digits, and special symbols. This reduces noise, ensuring the model focuses on meaningful patterns for more accurate sentiment predictions.

Accented Character Handling

Accented characters were replaced with their closest English equivalents using Unicode normalization (NFKD). This decomposes characters into their base form, efficiently eliminating diacritics.

Standardizing text in this way enhances model performance by reducing variation in word forms, preventing misclassification. This step is especially valuable for multilingual datasets or user-generated content, ensuring greater consistency in sentiment analysis.

Text Normalization

Text normalization ensured uniformity by converting all text to lowercase and standardizing spacing. This step reduces variability, allowing the model to treat words like "Great" and "great" identically, minimizing vocabulary size and eliminating case sensitivity issues.

Consecutive spaces were reduced, and leading/trailing spaces removed to prevent formatting inconsistencies from affecting analysis. This simplifies the data, ensuring minor variations don't skew sentiment predictions, which is crucial for handling diverse text sources.

Spelling Correction

Spelling correction was applied to refine text quality and prevent misspelled words from affecting model performance. This reduces noise from typographical errors, improving sentiment accuracy.

The Speller class from the autocorrect library was chosen for its speed, making it suitable for large datasets. This step is especially important for user-generated content, where spelling inconsistencies are frequent.

2. Model Development

The sentiment analysis model combines an embedding layer with a bidirectional LSTM to capture contextual and sequential dependencies in text. This architecture enhances the model's ability to understand sentiment by leveraging information from both past and future words..

Layer Breakdown

- Embedding Layer:
  - Converts input tokens into dense vectors using pre-trained embeddings. These embeddings are non-trainable, preserving external knowledge and preventing overfitting.
- Bidirectional LSTM:
  - A 128-unit bidirectional LSTM processes text in both forward and backward directions, capturing comprehensive context. Dropout and recurrent dropout reduce overfitting by randomly deactivating units during training.
- Dropout Layer:
  - A 50% dropout rate is applied after the LSTM to further prevent overfitting and promote generalization..
- Dense Layer:
  - A fully connected layer with 64 units and ReLU activation introduces non-linearity, enhancing the model's capacity to learn complex features.
- Output Layer:
  - The final layer outputs three class probabilities (negative, neutral, positive) using softmax activation for multi-class classification.

Compilation

The model uses the Adam optimizer for efficient and adaptive learning. Sparse categorical cross-entropy is chosen for its suitability in multi-class classification tasks with integer labels.

This architecture balances performance and regularization, ensuring the model generalizes well without overfitting.

3. Analysis of Results

The model delivered strong performance across training and test phases, achieving stable and reliable results.

Training Performance

The highest accuracy of 68.92% was recorded at epoch 17, with a minimum loss of 0.70655. Consistent performance in later epochs reflects the model's ability to generalize well without overfitting. Dropout layers contributed to this stability by preventing excessive reliance on specific patterns.

Test Performance

The model achieved:

- Test Accuracy: 68.08%
- Test Loss: 0.7235

This aligns closely with validation results, indicating strong generalization to unseen data.

Classification Report

- Class 0 (Negative): F1-score 0.66
- Class 1 (Neutral): F1-score 0.63
- Class 2 (Positive): F1-score 0.75

The model excelled in identifying positive sentiments, while neutral predictions were slightly less accurate, reflecting the challenge of distinguishing overlapping sentiments.

The bidirectional LSTM effectively captured sentiment by leveraging context from both directions, ensuring strong classification of positive and negative emotions.

Pre-trained embeddings and dropout layers enhanced performance and prevented overfitting, contributing to consistent accuracy across training and test data.

While the model excelled in polarized sentiment detection, neutral class predictions were slightly less accurate, indicating potential for further refinement.