Text Preprocessing Techniques for Mining Digital Data

1 Methodology

1.1 Research Design

This study employs a quantitative experimental approach to evaluate the performance of transformer-based deep learning models for sentiment analysis. The research methodology follows a structured pipeline consisting of data preparation, model training, evaluation, and comparative analysis. The experimental design focuses on comparing different transformer architectures (BERT and RoBERTa) across various datasets to determine their efficacy in sentiment classification tasks.

1.2 Dataset Selection and Preparation

Three distinct datasets were utilized in this study to provide a comprehensive evaluation of model performance across different text domains:

- **Stanford Sentiment Treebank (SST)**: A widely-used benchmark dataset in sentiment analysis research, containing movie review excerpts annotated with fine-grained sentiment labels.
- **Reddit Dataset**: A collection of social media comments extracted from Reddit, reflecting more informal, user-generated content with sentiment annotations.
- **Combined Dataset**: An integrated dataset merging both SST and Reddit data to evaluate model performance on heterogeneous text sources.

The datasets were pre-processed and divided into training and validation sets using a stratified sampling approach to maintain class distribution. Each dataset was processed to ensure consistent formatting, with texts cleaned and standardized before being fed into the model training pipeline.

1.3 Model Architecture

This study implemented two state-of-the-art transformer-based architectures for sentiment analysis:

1.3.1 BERT (Bidirectional Encoder Representations from Transformers)

BERT represents a significant advancement in natural language processing due to its bidirectional contextual understanding of text. The implementation utilized the `bert-base-uncased` pre-trained model, which consists of:

- 12 transformer layers
- 768 hidden dimensions
- 12 attention heads
- 110 million parameters

The model was fine-tuned for the sentiment classification task by adding a classification layer on top of the pre-trained BERT encoder.

1.3.2 RoBERTa (Robustly Optimized BERT Pretraining Approach)

RoBERTa builds upon BERT's architecture with modified key hyperparameters and training methodology. This study employed the `roberta-base` variant, which maintains the same structural characteristics as BERT but differs in pre-training approach:

- 12 transformer layers
- 768 hidden dimensions
- 12 attention heads
- 125 million parameters

RoBERTa removes BERT's next-sentence prediction objective and is trained with larger batches, more data, and dynamic masking patterns, potentially offering improved performance.

1.4 Training Procedure

1.4.1 Model Configuration

Six distinct model configurations were trained and evaluated:

1. BERT base model trained on SST dataset
2. BERT base model trained on Reddit dataset
3. BERT base model trained on the combined dataset
4. RoBERTa base model trained on SST dataset
5. RoBERTa base model trained on Reddit dataset
6. RoBERTa base model trained on the combined dataset

This experimental design enables cross-comparison between model architectures and data domains.

1.4.2 Hyperparameter Selection

The training process utilized the following hyperparameters, optimized based on preliminary experiments and prior research:

The AdamW optimizer was selected for its effectiveness in fine-tuning transformer models, combining the benefits of Adam optimization with proper weight decay regularization.

1.4.3 Training Infrastructure

All models were trained using PyTorch and the Hugging Face Transformers library. Training was conducted on CUDA-compatible GPU hardware to accelerate the fine-tuning process. Gradient clipping (max norm of 1.0) was applied to prevent exploding gradients.

1.4.4 Token Processing

Text inputs were tokenized using architecture-specific tokenizers:

- BERT models: WordPiece tokenization from `bert-base-uncased`
- RoBERTa models: Byte-Pair Encoding from `roberta-base`

Sequences were truncated or padded to a fixed length of 128 tokens to ensure consistent input dimensions, balancing computational efficiency with minimal information loss.

1.5 Evaluation Metrics

1.5.1 Primary Metrics

- **Accuracy**: The proportion of correctly classified instances across all classes.
- **Macro-averaged F1 Score**: The harmonic mean of precision and recall calculated independently for each class and then averaged.

1.5.2 Training Efficiency Metrics

- **Training Loss**: The cross-entropy loss on the training data, monitored across epochs.
- **Validation Loss**: The cross-entropy loss on the validation data, used to detect overfitting.

1.6 Model Selection and Validation

The best-performing model for each configuration was selected based on the highest macro-F1 score achieved on the validation set during training. Checkpoints were saved after each training epoch, with the best-performing checkpoint (based on validation F1 score) retained as the final model. This approach mitigates overfitting and ensures optimal performance.

1.7 Implementation Details

The training pipeline was implemented in Python using PyTorch and the Transformers library. An object-oriented design was adopted, encapsulating logic within a `ModelTrainer` class for modularity and extensibility.

Key features include:

- Automatic label mapping
- Progressive logging of training statistics
- Automated checkpointing and best model selection
- Comprehensive metrics tracking and JSON-based serialization

1.8 Experimental Controls

To ensure fair comparison between model architectures and datasets, several controls were implemented:

- Consistent random seeds for reproducibility
- Identical preprocessing steps across datasets
- Uniform evaluation metrics and procedures
- Standardized hardware environment for all training runs

1.9 Ethical Considerations

The datasets used in this research were publicly available and anonymized. For the Reddit dataset, all personally identifiable information had been removed prior to analysis. The research adheres to ethical standards for data usage and model development.

Hyperparameter	**BERT Value**	**RoBERTa Value**
Learning Rate	2e-5	2e-5
Batch Size	16	16
Maximum Sequence Length	128	128
Weight Decay	0.01	0.01
Training Epochs	4	3
Optimizer	AdamW	AdamW
Learning Rate Schedule	Linear with warmup	Linear with warmup

Table 1: Hyperparameters used for model training