# IMDB Sentiment Analysis: Technical Report

## Dataset Overview

The project utilizes the IMDB Movie Reviews dataset, a widely-used benchmark for sentiment analysis. Key characteristics: - **Size**: 50,000 movie reviews (25,000 training, 25,000 testing) - **Balance**: Equal distribution of positive and negative reviews - **Content**: Detailed movie reviews with rich textual information - **Labels**: Binary classification (Positive/Negative) - **Advantages**: - High-quality, human-generated content - Natural language variation - Real-world application context

## Technical Implementation

### Text Preprocessing Pipeline

1. **Case Normalization**
   * Convert all text to lowercase
   * Why: Ensures consistency, reduces vocabulary size
2. **Special Character Handling**
   * Remove non-alphabetic characters
   * Why: Reduces noise, focuses on meaningful words
3. **Advanced Text Processing**
   * **Tokenization**: Using NLTK’s word\_tokenize
   * **Lemmatization**: WordNetLemmatizer with POS tagging
   * Why:
     + Reduces word variations to base form
     + Maintains semantic meaning
     + More accurate than simple stemming
4. **Feature Engineering**
   * TF-IDF Vectorization
   * Parameters:
     + max\_features=5000: Optimal vocabulary size
     + ngram\_range=(1,2): Captures word pairs
   * Why:
     + Better than simple bag-of-words
     + Considers word importance
     + Captures phrase-level patterns

### Model Architecture

## Models and Implementation Approaches

### 1. Traditional ML Approach (sentiment\_analyzer.py)

* **Preprocessing**:
  + Case normalization
  + Special character removal
  + Lemmatization with POS tagging
* **Feature Engineering**: TF-IDF Vectorization
* **Models**:
  + Logistic Regression
  + Support Vector Machine (Linear Kernel)
* **Hyperparameter Optimization**: GridSearchCV

### 2. Deep Learning Approach (imdb\_trainer.ipynb)

* **Model**: BERT (bert-base-uncased)
* **Why BERT?**:
  + Superior performance on sentiment tasks
  + Pre-trained on large text corpus
  + Effective at capturing nuanced emotions
  + Uncased version suitable for sentiment analysis
* **Implementation Details**:
  + Max sequence length: 128 tokens
  + Batch size: 64
  + Learning rate: 2e-5
  + Training epochs: 15
  + Weight decay: 0.01
  + Warmup steps: 500

## Key Observations and Results

### Traditional ML Approach

1. **Preprocessing Impact**
   * Lemmatization improved semantic understanding
   * Bigram features captured phrase-level sentiment
   * Balanced class weights enhanced minority class prediction
2. **Model Performance**
   * Fast training and inference
   * Good baseline performance
   * Computationally efficient

### BERT Implementation

1. **Advantages**
   * Better handling of context
   * Captures subtle sentiment nuances
   * No manual feature engineering needed
2. **Trade-offs**
   * Longer training time
   * Higher computational requirements
   * More complex implementation

### Production Considerations

1. **Model Selection**
   * Traditional ML: Suitable for quick deployment, limited resources
   * BERT: Better for accuracy-critical applications
2. **Optimization Strategies**
   * Parameter grid optimization
   * Early stopping
   * Learning rate scheduling

## Conclusions

The project demonstrates two complementary approaches to sentiment analysis:

1. **Traditional ML Pipeline**
   * Efficient preprocessing
   * Fast training and inference
   * Good baseline performance
2. **BERT-based Deep Learning**
   * Superior accuracy potential
   * Better context understanding
   * Higher resource requirements

The choice between approaches depends on specific requirements: - Speed vs. Accuracy - Resource constraints - Deployment environment

### Final Thoughts

The project successfully demonstrates practical implementation of NLP techniques for sentiment analysis, with a focus on balancing performance and efficiency. The modular design allows for future enhancements and adaptations to different domains.