Sentiment Analysis Tool

Session: 2021 – 2025

Submitted by:

Muhammad Hamza 2021-CS-178

Supervised by:

Dr Usman Ghani

Department of Computer Science

University of Engineering and Technology

Lahore Pakistan

Contents

1. Project Overview

Sentiment analysis, also known as opinion mining, is a natural language processing (NLP) technique used to determine the sentiment or emotional tone behind a piece of text. It has applications in various domains such as marketing, customer service, and urban planning. This project aims to develop a sentiment analysis tool that utilizes two popular models, VADER and BERT, to analyze user-input text and provide a comparative analysis of their results. By combining these models, the tool offers insights into their performance and suitability for different contexts, bridging the gap between lightweight and deep learning-based sentiment analysis approaches.

Technologies Used

Programming Language

- Python: Used for implementing the logic and designing the GUI.

Libraries and Tools

1. **Tkinter**: For creating the graphical user interface.
2. **nltk (Natural Language Toolkit)**: Provides the VADER sentiment analysis model.
3. **transformers**: A library developed by Hugging Face to use pre-trained models like BERT.
4. **VADER**: A lexicon and rule-based sentiment analysis tool optimized for social media.
5. **BERT**: A deep learning-based sentiment analysis model trained on large datasets for contextual understanding.
6. **messagebox**: A Tkinter component for displaying error messages.

2. Research Gap

Identified Gap

Existing sentiment analysis tools often focus on using a single model, such as VADER or BERT, for sentiment classification. However, these tools lack:

- A comparative analysis between lightweight models (e.g., VADER) and advanced models (e.g., BERT) for longer and complex textual inputs.
- Insights into how different models perform on varying sentiment polarities (positive, neutral, negative).

Novelty

This project addresses this gap by:

1. Combining VADER and BERT sentiment analysis tools in a single application.
2. Providing a side-by-side comparison of sentiment predictions and detailed polarity scores.
3. Enabling users to analyze longer text inputs while adhering to the limitations of each model (e.g., BERT's 512-character limit).

---

3. Metrics and Literature Review

Metrics

- **Sentiment Polarity Scores**:
  - Positive, Neutral, Negative scores (VADER).
  - Prediction Labels and Confidence Scores (BERT).
- **Comparison Metric**: Identifying overlaps and discrepancies between VADER and BERT outputs for reliability and accuracy.

Literature Review

1. **Hutto & Gilbert (2014)**: Introduced VADER for efficient sentiment analysis in social media contexts.
2. **Devlin et al. (2018)**: Developed BERT, which revolutionized NLP tasks with a deep bidirectional transformer model.
3. **Comparative Studies**: Highlight limitations of lexicon-based models (like VADER) in nuanced contexts and emphasize BERT's strength in understanding contextual semantics.

---

4. Comparison

Results Comparison

The project compares:

- **VADER Scores**:
  - Lightweight and faster but less nuanced.
  - Best suited for short or informal text inputs.
- **BERT Predictions**:
  - More context-aware and accurate but computationally intensive.
  - Performs better on longer and grammatically complex sentences.

---

5. Documentation and References

Approach

1. **Tool Selection**:
    - VADER for its simplicity and speed.
    - BERT for its contextual understanding and advanced predictions.
2. **Integration**:
    - Implemented using Python and Tkinter for GUI development.
    - Utilized pre-trained models for sentiment analysis (NLTK's VADER and Hugging Face's BERT pipeline).

Results

- Real-time sentiment analysis using user-provided text.
- Comprehensive insights with both VADER and BERT results.

Findings

- VADER excels in processing short, informal text.
- BERT provides superior accuracy for nuanced sentiment detection.

References

1. Hutto, C. J., & Gilbert, E. (2014). *VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text*.
2. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*.
3. Hugging Face Transformers Documentation: https://huggingface.co/transformers/

---

6. System Requirements

Software Requirements

- Python 3.7 or higher
- Required libraries:
    - nltk
    - transformers

Hardware Requirements

- 2 GHz or faster processor
- At least 4 GB of RAM

- 50 MB of free disk space

---

7. Setup and Installation

1. Install Python

Download and install Python 3.7 or higher from python.org.

2. Install Required Libraries

Run the following commands to install the required libraries:

```
pip install nltk transformers
```

3. Download NLTK Data

Run the following command to download the VADER lexicon:

```
import nltk
nltk.download("vader_lexicon")
```

4. Run the Application

Save the provided Python script as `sentiment_analysis_tool.py` and execute it:

```
python sentiment_analysis_tool.py
```

---

8. Application Workflow

1. GUI Components

- **Input Section**: A text box where the user enters the paragraph to analyze.
- **Analyze Button**: A button that triggers the sentiment analysis process.
- **Output Section**: Displays the results of VADER and BERT sentiment analyses.

2. Sentiment Analysis Process

- **Step 1**: The user enters a paragraph into the input text box.
- **Step 2**: When the "Analyze Sentiment" button is clicked, the program performs the following:

    - **VADER Analysis**: Calculates sentiment scores for positive, neutral, and negative sentiments and determines the overall sentiment based on these scores.
    - **BERT Analysis**: Classifies the text as positive, negative, or neutral using a pre-trained BERT model.
- **Step 3**: Displays the analysis results in the output text box.

---

9. Code Explanation

1. Import Libraries

```
import tkinter as tk
from tkinter import Text, messagebox, filedialog
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from transformers import pipeline
import nltk
```

- **Tkinter**: Creates the GUI.
- **VADER and BERT**: Handle the sentiment analysis.

2. Download NLTK Data

```
nltk.download("vader_lexicon")
```

- Ensures that the VADER lexicon is available for analysis.

3. Initialize Models

```
vader_analyzer = SentimentIntensityAnalyzer()
bert_analyzer = pipeline("sentiment-analysis")
```

- **vader_analyzer**: Performs lexicon-based sentiment analysis.
- **bert_analyzer**: Uses a transformer-based BERT model for contextual sentiment classification.

4. Analyze Sentiment Function

```
def analyze_sentiment():
    input_text = input_textbox.get("1.0", tk.END).strip()
    if not input_text:
        messagebox.showerror("Error", "Please enter a paragraph to analyze!")
        return

    vader_scores = vader_analyzer.polarity_scores(input_text)
    vader_result = max(vader_scores, key=vader_scores.get)

    bert_result = bert_analyzer(input_text[:512])
```

```
    output_textbox.delete("1.0", tk.END)
    output_textbox.insert(tk.END, "=== Sentiment Analysis Results ===\n\n")
    output_textbox.insert(tk.END, f"VADER Result: {vader_result} (Score:
{vader_scores[vader_result]:.2f})\n")
    output_textbox.insert(tk.END, f"BERT Result: {bert_result[0]['label']}
(Score: {bert_result[0]['score']:.2f})\n\n")
    output_textbox.insert(tk.END, "=== VADER Scores ===\n")
    output_textbox.insert(tk.END, f"Positive: {vader_scores['pos']:.2f}\n")
    output_textbox.insert(tk.END, f"Neutral: {vader_scores['neu']:.2f}\n")
    output_textbox.insert(tk.END, f"Negative: {vader_scores['neg']:.2f}\n")

    output_textbox.insert(tk.END, "\n=== Research Insights ===\n")
    output_textbox.insert(tk.END, "Identified Gap: Lack of comparative
sentiment analysis using VADER and BERT for long text inputs.\n")
    output_textbox.insert(tk.END, "Evaluation Metrics: Sentiment polarity
scores, prediction accuracy comparison.\n")
```

- Retrieves the input text, processes it using both sentiment analysis models, and displays the results in the output box.

5. Save Results Function

```
def save_results():
    output_text = output_textbox.get("1.0", tk.END).strip()
    if not output_text:
        messagebox.showerror("Error", "No analysis results to save!")
        return

    file_path = filedialog.asksaveasfilename(defaultextension=".txt",
                                             filetypes=[("Text Files",
"*.txt"), ("All Files", "*.*")])
    if file_path:
        with open(file_path, "w", encoding="utf-8") as file:
            file.write(output_text)
        messagebox.showinfo("Success", "Results saved successfully!")
```

- Allows users to save the sentiment analysis results to a file.

6. Create the GUI

```
app = tk.Tk()
app.title("Sentiment Analysis Tool")
app.geometry("600x600")

# Input Section
tk.Label(app, text="Enter Paragraph:", font=("Arial", 12)).pack(pady=10)
input_textbox = Text(app, height=10, width=70)
input_textbox.pack()

# Analyze Button
analyze_button = tk.Button(app, text="Analyze Sentiment",
command=analyze_sentiment, bg="blue", fg="white", font=("Arial", 12))
analyze_button.pack(pady=10)
```

```
# Save Button
save_button = tk.Button(app, text="Save Results", command=save_results,
bg="green", fg="white", font=("Arial", 12))
save_button.pack(pady=5)

# Output Section
tk.Label(app, text="Sentiment Analysis Results:", font=("Arial",
12)).pack(pady=10)
output_textbox = Text(app, height=15, width=70, state="normal", bg="#f0f0f0")
output_textbox.pack()

app.mainloop()
```

- Defines the layout and interactive elements of the application.

Example Input and Output

Input Paragraph

Animals in China

This paragraph uses a topic sentence to state the main idea, supporting sentences to provide more specific information, and a concluding sentence to restate the topic:

Topic sentence: There are many different kinds of animals that live in China.

Supporting sentences: Tigers and leopards live in the forests in the north, monkeys swing in the trees in the jungles, elephants walk through the brush, and camels live in the deserts.

Concluding sentence: Lots of different kinds of animals make their home in China

Output

VADER Result: neu (Score: 0.85)

BERT Result: NEGATIVE (Score: 0.91)

=== VADER Scores ===

Positive: 0.15

Neutral: 0.85

Negative: 0.00

=== Research Insights ===

Identified Gap: Lack of comparative sentiment analysis using VADER and BERT for long text inputs.

Evaluation Metrics: Sentiment polarity scores, prediction accuracy comparison.

---

10. Future Enhancements

- **Integration with Social Media APIs**: Analyze real-time tweets or posts.
- **Data Visualization**: Add graphs to display sentiment trends over time.

---

11. Conclusion

The Sentiment Analysis Tool showcases the effective integration of both rule-based and deep learning-based NLP techniques to extract valuable insights from text data. By leveraging these advanced methodologies, the tool provides accurate sentiment analysis, enabling businesses, researchers, and policymakers to make data-driven decisions. With its ability to efficiently assess public sentiment, the tool serves as a vital asset for understanding trends, improving customer engagement, and shaping informed strategies. This combination of traditional and modern approaches ensures comprehensive, reliable results that can have a significant impact across various sectors.