© June 2026 | IJIRT | Volume 13 Issue 1 | ISSN: 2349-6002

Transformer Based Sentiment Analysis System

Nandini J[1], Dr Sukesha H A[2], Dr Krishna Kumar P R[3]
[1]*MTech Student, Department of CSE, SEA College of Engineering & Technology,Bangalore,India*
[2]*Professor, Department of CSE(IOT+BC+CB), SEA College of Engineering & Technology,Bangalore,India*
[3]*Professor, Department of CSE, SEA College of Engineering & Technology,Bangalore,India*

***Abstract*—Sentiment Analysis, also known as opinion mining, is a key application of Natural Language Processing (NLP) that focuses on identifying and classifying emotions expressed in textual data. With the rapid growth of digital platforms such as social media, e-commerce websites, and online forums, a large volume of user-generated content is produced every day. Analyzing this data manually is inefficient, which makes automated sentiment analysis systems essential. Traditional methods such as Naive Bayes and Logistic Regression often fail to capture contextual meaning, leading to limited accuracy.**

This project proposes a sentiment analysis system using a pre-trained transformer-based model (RoBERTa), which effectively understands contextual relationships in text. The system performs preprocessing of input text, tokenization using the model tokenizer, and directly utilizes the pre-trained model for sentiment prediction without additional training. The model classifies text into positive, negative, or neutral categories based on contextual understanding.

The proposed approach improves accuracy and performance by leveraging large-scale pre-training, making it suitable for applications such as customer feedback analysis, product reviews, and social media monitoring.

I. INTRODUCTION

1.1 Background

In recent years, the rapid growth of digital platforms such as social media, e-commerce websites, blogs, and online forums has resulted in an enormous increase in the volume of textual data generated by users. People frequently share their opinions, reviews, and experiences online, creating a valuable source of information for organizations and researchers.

However, manually processing such large-scale textual data is time-consuming and inefficient. This has led to the development of automated techniques under Natural Language Processing (NLP), particularly sentiment analysis, which focuses on identifying and classifying the emotional tone of text. Sentiment analysis enables systems to extract subjective information, such as opinions and attitudes, and categorize them into predefined classes like positive, negative, or neutral. With advancements in deep learning and transformer-based models, sentiment analysis has become more accurate and capable of capturing complex language patterns and contextual meanings.

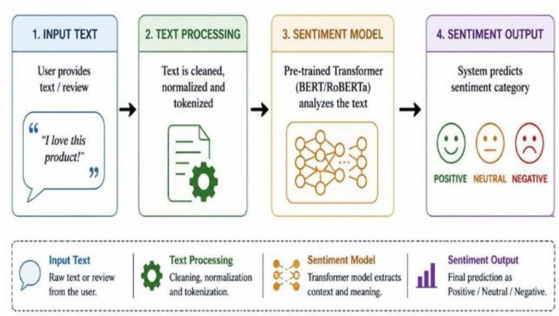

Fig 1.1 Basic Sentiment Analysis Process

1.2 Problem Statement

Traditional sentiment analysis techniques such as Naive Bayes, Support Vector Machines, and Logistic Regression have been widely used for text classification tasks. However, these approaches primarily

rely on bag-of-words or term frequency-based representations, which treat words as independent features and ignore the order and context in which they appear. As a result, such models struggle to capture the semantic relationships between words, leading to reduced accuracy when dealing with complex sentences, negations, sarcasm, and context-dependent meanings.

Furthermore, these methods require extensive manual feature engineering, including text preprocessing, selection of relevant features, and tuning of parameters, which increases the complexity of the system. They are also limited in handling large-scale and diverse datasets where language usage varies significantly. Due to these limitations, there is a need for more advanced approaches that can understand contextual information effectively. Transformer-based models like BERT address these challenges by learning deep contextual representations of text, thereby improving the performance and reliability of sentiment analysis systems.

1.3 Objectives

The primary objective of this project is to develop a robust Sentiment Analysis System using BERT (Transformer Model) and Natural Language Processing techniques. The specific objectives include:

- To utilize a pre-trained transformer model for sentiment analysis
- To clean and normalize input text data
- To tokenize input text using the model tokenizer
- To classify text into positive, negative, and neutral sentiments
- To develop a real-time sentiment analysis system

1.4 Scope

This project focuses on sentiment classification of textual data using deep learning techniques, specifically the BERT model. The scope of the project includes:

- Analysis of textual data from various domains such as social media, reviews, and feedback
- Application of transformer-based models for improved contextual understanding
- Classification of sentiments into positive, negative, and neutral categories
- Handling large-scale datasets efficiently using deep learning techniques
- Implementation of a scalable and adaptable sentiment analysis system

1.5 Applications

Sentiment Analysis has a wide range of real-world applications across various domains:

- E-commerce: Helps analyze customer reviews and improve product quality
- Social Media Monitoring: Tracks public opinion on trending topics
- Business Intelligence: Assists companies in decision-making
- Customer Support: Identifies customer dissatisfaction quickly
- Politics: Analyses public opinion on policies and elections
- Healthcare: Evaluates patient feedback and service quality

II. SENTIMENT ANALYSIS AND NLP

2.1 Introduction

Natural Language Processing (NLP) is a branch of Artificial Intelligence that focuses on enabling machines to understand, interpret, and generate human language. It combines linguistics, computer science, and machine learning to process textual data.
Sentiment Analysis is one of the most important applications of NLP, which involves identifying the emotional tone behind a body of text. It helps determine whether the sentiment expressed is positive, negative, or neutral.

2.2 Types of Sentiment Analysis

1. Document-Level Sentiment Analysis
   - Analyses the overall sentiment of an entire document
   - Suitable for long reviews
2. Sentence-Level Sentiment Analysis
   - Determines sentiment at sentence level
   - Useful for detailed analysis
3. Aspect-Level Sentiment Analysis
   - Focuses on specific aspects or features
   - Example: “Camera is good, but battery is poor”

2.3 Approaches to Sentiment Analysis

2.3.1 Rule-Based Approaches

Rule-based methods rely on predefined linguistic rules and sentiment lexicons (dictionaries of words with associated sentiment scores). These systems analyze the presence of positive or negative words and apply grammatical rules to determine sentiment.

Advantages**:**
- Simple to implement
- Does not require training data

Limitations:
- Cannot handle complex language structures
- Limited scalability and adaptability

2.3.2 Machine Learning Approaches

Machine learning methods use labeled datasets to train models that can classify sentiments. Common algorithms include:
- Naive Bayes
- Support Vector Machines (SVM)
- Logistic Regression

These models require feature extraction techniques such as Bag-of-Words and TF-IDF to convert text into numerical form.

Advantages:
- Better performance than rule-based systems
- Can generalize from training data

Limitations:
- Requires feature engineering
- Struggles with context and semantics

2.3.3 Deep Learning Approaches

Deep learning methods use neural networks to automatically learn features from text data. Common architectures include:
- Recurrent Neural Networks (RNN)
- Long Short-Term Memory (LSTM)
- Convolutional Neural Networks (CNN)

These models improve performance by capturing sequential and contextual information in text.

Advantages:
- High accuracy
- Reduced need for manual feature engineering

Limitations:
- Requires large datasets
- Computationally expensive

2.4 Challenges in Sentiment Analysis

Despite advancements, sentiment analysis faces several challenges:
- Ambiguity in Language: Words can have multiple meanings depending on context
- Sarcasm and Irony: Difficult to detect using traditional models
- Negation Handling: Phrases like “not good” can confuse models
- Domain Dependency: Words may have different sentiments in different contexts
- Multilingual Issues: Handling multiple languages adds complexity

Addressing these challenges requires models capable of deep contextual understanding, such as transformer-based architectures.

2.5 Introduction to Transformer Models

Transformer models represent a significant advancement in NLP. Unlike traditional sequential models, transformers process all words in a sentence simultaneously using a mechanism called self-attention.

This allows the model to understand relationships between words regardless of their position in the text.

The key component of transformers is the attention mechanism, which assigns importance to different

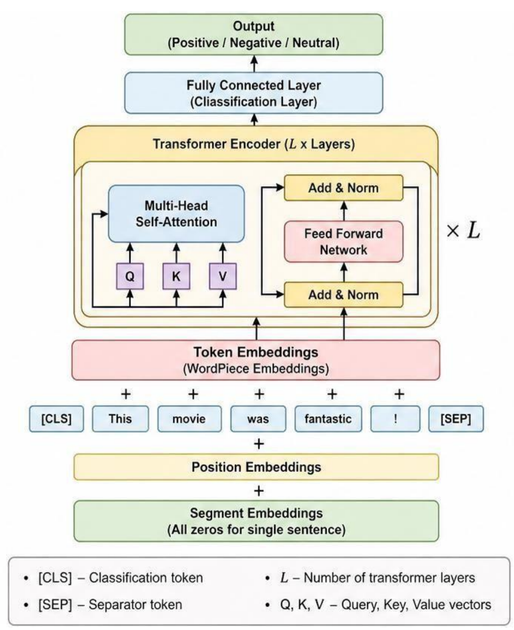

Fig 2.1 Architecture of Roberta model

words in a sentence while processing a specific word. This enables the model to capture long-range dependencies and contextual information more effectively.

Transformers consist of encoder and decoder blocks.

In many NLP tasks, including sentiment analysis, only the encoder part is used to extract meaningful representations of text.

2.6 BERT (Bidirectional Encoder Representations from Transformers)

2.6.1 Overview

BERT is a pre-trained transformer-based model developed by Google for NLP tasks. It is designed to understand the context of words in a sentence by considering both the left and right surroundings (bidirectional context).

2.6.2 Key Features

- Bidirectional context understanding
- Pre-trained on large text corpora
- Fine-tuning capability for specific tasks
- State-of-the-art performance in many NLP tasks

2.6.3 Working Principle of BERT

BERT processes input text by converting it into a structured format consisting of tokens and embeddings. The input representation includes:

- Token Embeddings: Represent individual words or sub words
- Segment Embeddings: Distinguish between sentences
- Positional Embeddings: Indicate the position of words in a sequence

These embeddings are passed through multiple transformer encoder layers, where self-attention mechanisms analyze relationships between words. The output is a contextualized representation of each word in the sentence.

2.6.4 Pre-training Tasks in BERT

BERT is pre-trained on large-scale unlabeled text data using two important tasks that help the model learn deep contextual representations of language:

Masked Language Modeling (MLM)**:-** In this task, a certain percentage (typically 15%) of words in a sentence are randomly masked, and the model is trained to predict the original words based on the surrounding context. This allows BERT to learn bidirectional relationships, as it considers both left and right context while making predictions. Unlike traditional language models, which process text in a single direction, MLM enables a more comprehensive understanding of sentence structure and meaning.

Next Sentence Prediction (NSP)**:-** In this task, the model is given pairs of sentences and learns to predict whether the second sentence logically follows the first one. This helps BERT understand sentence relationships, coherence, and context across multiple sentences. NSP is particularly useful for tasks such as question answering and natural language inference, where understanding the relationship between sentences is crucial.

These two pre-training objectives enable BERT to develop a strong understanding of language patterns, grammar, and context before being fine-tuned for specific tasks like sentiment analysis. As a result, the model requires less labeled data during fine-tuning and achieves higher performance compared to models trained from scratch.

2.6.5 Fine-Tuning for Sentiment Analysis

After pre-training, BERT is fine-tuned for specific tasks like sentiment analysis by adding a classification layer on top of the model. The entire model is then trained on a labeled dataset to classify text into sentiment categories such as positive, negative, and neutral.

During fine-tuning, the input text is first tokenized and passed through the pre-trained BERT model to obtain contextual embeddings. The representation of the special [CLS] token, which captures the overall meaning of the sentence, is then fed into a fully connected (dense) layer followed by a SoftMax activation function to produce probability scores for each sentiment class. The model parameters are slightly adjusted using backpropagation to minimize classification error.

III. PROPOSED METHODOLOGY

3.1 Introduction

The proposed system automatically classifies sentiments from textual data using a pre-trained transformer model (RoBERTa). It improves accuracy by capturing the contextual meaning of words rather than relying only on word frequencies. The system efficiently processes unstructured text and converts it into meaningful insights.

In this approach, Natural Language Processing (NLP) techniques are used to clean and normalize the input data before processing. The pre-trained model is directly used for sentiment prediction without additional training, enabling effective understanding of contextual relationships and sentiment variations.

3.2 Existing System

The existing sentiment analysis systems primarily rely on traditional approaches such as manual analysis, rule-based methods, and basic machine learning algorithms. Manual analysis is time-consuming and subjective, while lexicon-based methods use predefined dictionaries to classify sentiment, often lacking flexibility.

Machine learning techniques such as Naive Bayes, Support Vector Machines (SVM), and Logistic Regression improved performance by learning from labeled data. However, these models treat words independently and fail to capture contextual relationships. As a result, they struggle with negation, sarcasm, and domain-specific language, leading to reduced accuracy and poor generalization.

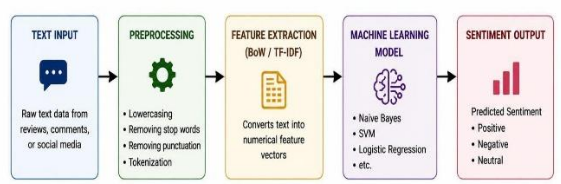

Fig 3.1 Existing Sentiment Analysis System

3.3 Limitations of Existing System

- Time-consuming manual analysis
- Low accuracy in rule-based systems
- Poor handling of sarcasm and context
- Inability to capture semantic relationships between words
- Not scalable for large datasets
- Limited adaptability to new domains

3.4 Proposed System

The proposed system uses a transformer-based deep learning model. Specifically, it utilizes the cardiffnlp/twitter-roberta-base-sentiment pre-trained model from Hugging Face, which is trained on large-scale Twitter data and supports three-class sentiment classification (positive, negative, neutral).

The system uses a pre-trained RoBERTa model without additional training. The model is directly used for sentiment prediction. During processing, the input text is tokenized using the RoBERTa tokenizer and converted into embeddings that capture contextual meaning. The pre-trained model then classifies the input text into sentiment categories such as positive, negative, or neutral.

Once trained, the system can accurately predict the sentiment of new, unseen text inputs. The output is presented to the user in a clear and understandable format. The proposed system is scalable and can handle large volumes of data efficiently. It can also be extended for real-time sentiment analysis and integrated with web or mobile applications.

Key Features of Proposed System

- Automated sentiment classification using deep learning
- High accuracy through contextual understanding of text
- Reduced need for manual feature engineering
- Efficient preprocessing and tokenization using BERT
- Scalable and adaptable system
- Supports real-time sentiment analysis

3.5 System Architecture

The system architecture is designed in a modular manner for efficient processing of textual data. The system includes:

- Input Module: Accepts user text
- Preprocessing Module: Cleans and normalizes text
- Tokenization Module: Converts text into tokens using RoBERTa tokenizer
- Model Module: Processes input using a pre-trained transformer model
- Classification Module: Predicts sentiment
- Prediction Module: Displays the final result

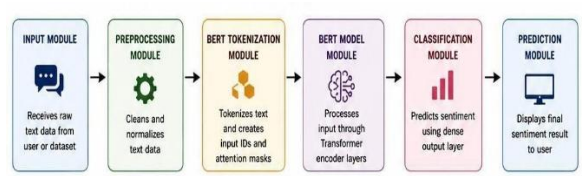

Fig 3.2 System Architecture of Proposed Sentiment Analysis System

3.6 Working Principle

1. The system takes raw text input such as reviews,

comments, or tweets from the user or dataset.

2. The input text is passed to the preprocessing stage, where it is cleaned by:

- Converting text to lowercase
- Removing stop words
- Removing punctuation and special characters
- Tokenizing the text into words

3. The cleaned text is then processed using the BERT tokenizer, which converts it into input IDs and attention masks.
4. The tokenized data is fed into the pre-trained BERT model.
5. The model processes the text using transformer layers to generate contextual embeddings.
6. The [CLS] token representation is passed to a classification layer.
7. The model predicts the sentiment of the text based on learned patterns.
8. The output is classified into:

- Positive
- Negative
- Neutral

9. Finally, the predicted sentiment is displayed to the user in an understandable format.

3.7 Flowchart of the Proposed Model

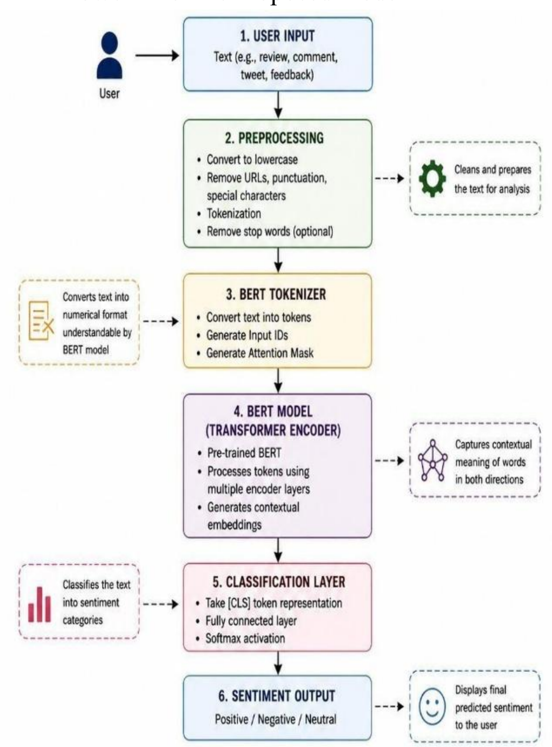

Fig. 3.3 Flowchart of Sentiment Analysis System

3.8 Pre-Processing Steps

The purpose of pre-processing is to clean and transform the input text into a structured format suitable for machine learning algorithms. Effective pre-processing improves the accuracy and efficiency of the model.

The following steps are performed during the pre-processing phase:

- Lowercasing:

All characters in the text are converted to lowercase to maintain uniformity and avoid duplication of words with different cases.

- Removal of Stop words:

Commonly used words such as “is”, “the”, “and”, etc., which do not carry significant meaning, are removed to reduce unnecessary data.

- Tokenization:

The text is split into smaller units called tokens (words or terms), which helps in analysing individual components of the text.

- Removal of Punctuation and Special Characters:

Unnecessary symbols such as punctuation marks and special characters are removed to ensure clean and consistent data.

- Stemming/Lemmatization:

Words are reduced to their base or root form (e.g., “running” to “run”), which helps in minimizing variations of the same word.

- Removal of Numbers:

Numeric values are removed unless they contribute to the meaning of the text.

- Handling Emojis and Slang (if applicable):

Emojis and informal expressions are converted into meaningful text representations to better capture sentiment.

3.9 Algorithm

Algorithm: Sentiment Analysis using Pre-trained Roberta Model

Input: User text

Output: Sentiment (Positive / Negative / Neutral)

Steps:

1. Accept text input from user via web interface.
2. Preprocess text (optional cleaning).
3. Load pre-trained RoBERTa model using Hugging Face pipeline.

4. Pass input text to the model.
5. Obtain predicted label and confidence score.
6. Apply custom rule for low-confidence predictions.
7. Display sentiment result to user.

3.10 Advantages of the Proposed System

- No need for dataset or training
- Faster implementation using Hugging Face pipeline
- Real-time prediction using Flask API
- High accuracy due to pre-trained model
- Simple and scalable architecture

IV. IMPLEMENTATION

4.1. Technologies Used

- Python
- Flask
- Transformers (Hugging Face)
- Pre-trained Model: cardiffnlp/twitter-roberta-base-sentiment (RoBERTa)
- HTML, CSS
- JavaScript

4.2. Hardware Requirements

- RAM: Minimum 4GB
- Processor: Intel i3 or higher
- Storage: 2–5GB

4.3. Software Requirements

- Operating System: Windows/Linux/MacOS
- Programming Language: Python 3.x
- Libraries/Frameworks: Flask, Transformers (Hugging Face), PyTorch, NumPy
- IDE/Editor: VS Code

4.4. Code Description

- Frontend: HTML form to input text
- Backend: Flask handles requests
- Model: Pre-trained transformer for prediction

4.5 Appendix

Appendix A: Backend Code (app.py): -

```
from flask import Flask, render_template, request, jsonify from transformers import pipeline
app = Flask(__name__) # 3-class model classifier =
pipeline("sentiment-analysis",
model="cardiffnlp/twitter-roberta-base-sentiment",
tokenizer="cardiffnlp/twitter-roberta-base-sentiment")
@app.route("/") def home(): return
render_template("index.html") @app.route("/predict",
methods=["POST"]) def predict(): try:
data = request.get_json() text = data.get("text") result
= classifier(text)[0] # Label mapping label_map = {
"LABEL_0": "negative", "LABEL_1": "neutral",
"LABEL_2": "positive"}
sentiment = label_map[result["label"]] score =
result["score"]
# Custom rule: weak positive → neutral if sentiment
== "positive" and score < 0.85:sentiment = "neutral"
confidence = round(score * 100, 2) return jsonify({
"sentiment": sentiment, "confidence": confidence})
except Exception as e: print("Error:", e) return
    jsonify({"error": "Server error"}), 500 if _name==
    "_main_":
app.run(debug=False)
```

Appendix B: Frontend Code (index.html)

```
<!DOCTYPE html>
<html>
<head>
<title>AI Sentiment Analysis</title>
<link rel="stylesheet" href="{{ url_for('static',
filename='style.css') }}">
</head>
<body>
<div class="container">
<h2>Context-Aware Sentiment Analysis (AI)</h2>
<textarea id="inputText" rows="5"
placeholder="Enter your text"></textarea>
<button onclick="analyze()">Analyze
Sentiment</button>
<h3 id="result"></h3>
</div>
<script>
function analyze() {
let text =
document.getElementById("inputText").value;
if (text.trim() === "") {
document.getElementById("result").innerText =
"Please enter some text!"; return;}
fetch("/predict", { method: "POST", headers: {
"Content-Type": "application/json"},
body: JSON.stringify({ text: text })})
.then(response => response.json())
```

```
.then(data => {
document.getElementById("result").innerText =
"Sentiment: " + data.sentiment + " (" +
data.confidence + "%)";
})
.catch(error => {
document.getElementById("result").innerText =
"Error connecting to server!";});}
</script>
</body>
</html>
```

Appendix C: CSS Code (style.css): -

```
body {
font-family: Arial, sans-serif;
background: linear-gradient(to right, #667eea,
#764ba2); text-align: center;
color: white; margin: 0;
padding: 0;}
.container {
margin-top: 100px; background: rgba(0, 0, 0, 0.3);
padding: 30px;
border-radius: 15px; width: 50%;
margin-left: auto; margin-right: auto;}
h2 {margin-bottom: 20px;}
textarea { width: 80%;
padding: 10px; border-radius: 10px; border: none;
resize: none;
font-size: 16px;}
button {margin-top: 15px; padding: 10px 20px;
border: none; border-radius: 10px;
background-color: #00c9a7; color: white;
font-size: 16px; cursor: pointer;}
button:hover {
background-color: #00a389;}
#result {margin-top: 20px; font-size: 20px; font-
weight: bold;}
```

V. RESULTS AND DISCUSSION

5.1 Introduction

This chapter presents the results obtained from the developed context-aware sentiment analysis system. The system utilizes a pre-trained transformer-based model to classify input text into positive, negative, or neutral sentiment categories. The performance of the system is evaluated using sample user inputs.

5.2 Experimental Results

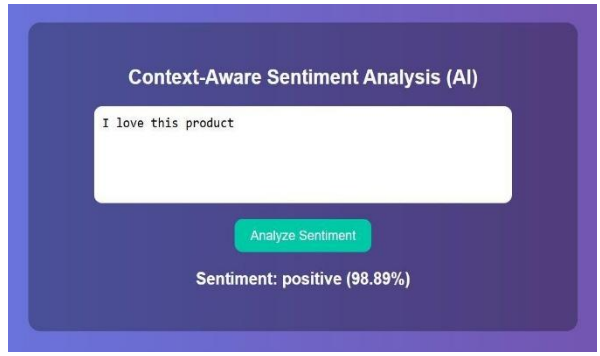

Fig 5.1 Sample Output – Positive Sentiment Prediction

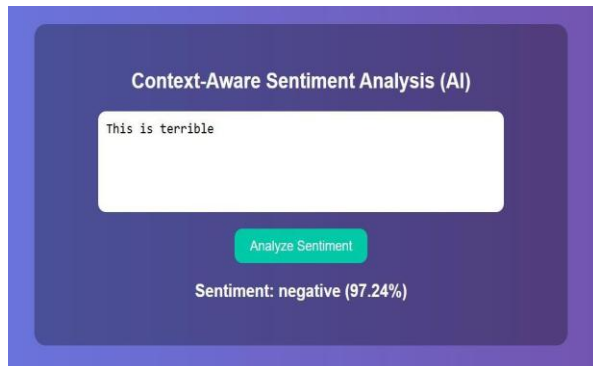

Fig 5.2 Sample Output – Negative Sentiment Prediction

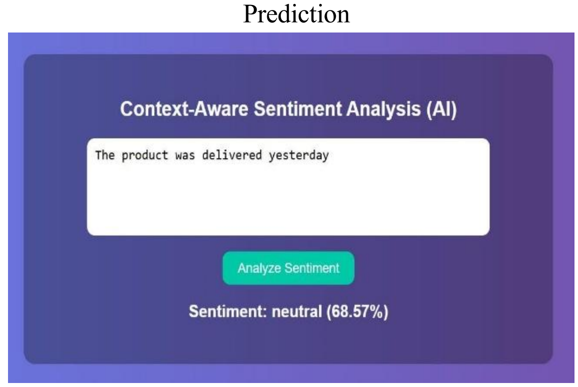

Fig 5.3 Sample Output – Neutral Sentiment Prediction

5.3 Discussion

The results indicate that the transformer-based model performs well in understanding contextual relationships within text. The system accurately classifies sentences with clear positive or negative sentiment.

Key Observations

- The model provides high accuracy for strongly expressed sentiments.
- Context-aware understanding improves

prediction quality.

- Neutral sentences are correctly identified when no emotional tone is present.
- Borderline sentences such as “The product is okay” may sometimes be classified as slightly positive due to real-world sentiment interpretation.

5.4 Advantages

- No need for custom dataset or training
- High accuracy due to pre-trained model
- Context-aware sentiment understanding
- Easy deployment using Flask

5.5 Limitations

- Requires internet for initial model download
- Slight delay in response time due to model size
- May misclassify mildly neutral or ambiguous sentences
- Performance depends on pre-trained data distribution

VI. CONCLUSION AND FUTURE WORK

6.1 Conclusion

This project presented a Context-Aware Sentiment Analysis System using Transformer-Based Natural Language Processing aimed at improving the accuracy and efficiency of sentiment classification. The proposed system integrates a Flask-based web application with a pre-trained transformer model to intelligently analyze and classify textual data into positive, negative, and neutral sentiments.

The findings demonstrate that the transformer-based approach can:

1. Understand contextual relationships within text effectively.
2. Eliminate the need for custom dataset training by using pre-trained models.
3. Provide high accuracy in sentiment prediction across various inputs.
4. Handle real-world language variations better than traditional methods.

Compared to conventional machine learning approaches, the transformer-based model provides a more robust and context-aware solution, making it suitable for real-time applications such as customer feedback analysis and social media monitoring.

6.2 Future Work

Future enhancements of the system may include:

- Implementing multilingual sentiment analysis
- Integrating with real-time data sources (social media platforms)
- Developing interactive dashboards for visualization
- Using advanced transformer models for improved accuracy
- Optimizing system performance for faster response time
- Extending the system for emotion and sarcasm detection

REFERENCES

[1] A. S. Talaat, “Sentiment analysis classification system using hybrid BERT models,” Journal of Big Data, vol. 10, no. 110, 2023.

[2] H. Xiao and L. Luo, “An Automatic Sentiment Analysis Method for Short Texts Based on Transformer-BERT Hybrid Model,” IEEE Access, vol. 12, pp. 93305–93317, 2024.

[3] Z. Li, C. Yang, and C. Huang, “A Comparative Sentiment Analysis of Airline Customer Reviews Using BERT and Its Variants,” Mathematics, vol. 12, no. 1, pp. 53, 2024.

[4] M. A. Jahin, M. S. H. Shovon, M. F. Mridha, et al., “A Hybrid Transformer and Attention-Based Recurrent Neural Network for Robust and Interpretable Sentiment Analysis of Tweets,” Scientific Reports, vol. 14, Article 24882, 2024.

[5] K. Aziz, D. Ji, P. Chakrabarti, et al., “Unifying Aspect-Based Sentiment Analysis BERT and Multi-Layered Graph Convolutional Networks for Comprehensive Sentiment Dissection,” Scientific Reports, vol. 14, Article 14646, 2024.

[6] E. Hashmi, S. Y. Yayilgan, and S. Shaikh, “Augmenting Sentiment Prediction Capabilities for Code-Mixed Tweets with Multilingual Transformers,” Social Network Analysis and Mining, vol. 14, no. 86, 2024.

[7] A. Bello, S.-C. Ng, and M.-F. Leung, “A BERT Framework to Sentiment Analysis of Tweets,” Sensors, vol. 23, no. 1, pp. 506, 2023.

[8] N. Cassee, A. Agaronian, E. Constantinou, N. Novielli, and A. Serebrenik, “Transformers and Meta-Tokenization in Sentiment Analysis for Software Engineering,” Empirical Software Engineering, vol. 29, no. 77, 2024.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” Proceedings of NAACL-HLT, 2019.
[10] Ashish Vaswani et al., “Attention Is All You Need,” Advances in Neural Information Processing Systems (NeurIPS), 2017.
[11] D. Yin, T. Meng, and K.-W. Chang, “SentiBERT: A Transferable Transformer-Based Architecture for Compositional Sentiment Semantics,” 2020.
[12] S. Alaparthi and M. Mishra, “Bidirectional Encoder Representations from Transformers (BERT): A Sentiment Analysis Odyssey,” 2020.
[13] H. Batra, N. S. Punn, S. K. Sonbhadra, and S. Agarwal, “BERT-Based Sentiment Analysis: A Software Engineering Perspective,” 2021.
[14] O. E. Ojo, H. T. Ta, A. Gelbukh, H. Calvo, O. O. Adebanji, and G. Sidorov, “Transformer-Based Approaches to Sentiment Detection,” 2023.
[15] H. Zhang and M. O. Shafiq, “Survey of Transformers and Towards Ensemble Learning Using Transformers for Natural Language Processing,” Journal of Big Data, vol. 11, no. 25, 2024.