Aspect Based Sentiment Analysis on Codemix Languages

Dr. Keerthi H M

Associate Professor,
B.E., M.Tech, Ph.D,
Dept. of Computer Science and Engg.
Malnad College of Engineering Hassan, India

Mr. Sudarshan K P
Dept of Computer Science and Engg.
Malnad College of Engineering
Hassan, India

Mr. Siddarth S Mallik
Dept of Computer Science and Engg.
Malnad College of Engineering
Hassan, India

Mr. Sumeeth S M
Dept of Computer Science and Engg.
Malnad College of Engineering
Hassan, India

Mr. Sumit Chandra
Dept of Computer Science and Engg.
Malnad College of Engineering
Hassan, India

Abstract

In the era of social media, where multilingual conversations are prevalent, analyzing code-mixed text poses unique challenges. This project presents a comparative analysis of sentiment analysis and aspect-based sentiment analysis on code-mixed data using advanced techniques like Large Language Models (LLM), BERT, and Naive Bayes.

Sentiment analysis categorizes text into positive, negative, or neutral sentiments, while aspect-based analysis identifies opinions on specific topics, such as "price" or "quality" in reviews. By focusing on code-mixed text, this study compares the effectiveness of each method in understanding sentiments and specific opinions, paving the way for improved applications in multilingual settings.

I. Introduction

Code-mixed text, where multiple languages blend in a single conversation, is common in social media. This project aims to identify sentiments and specific opinions (aspects) in such multilingual texts using comparative techniques.

The models analyzed include LLMs, BERT, and Naive Bayes. Sentiment analysis identifies general sentiments, while aspect-based sentiment analysis focuses on specific features. This study's focus on code-mixed data highlights the complexities of multilingual contexts and evaluates the models' performances under these conditions

II. Problem Statement

With the increasing prevalence of code-mixed languages, traditional sentiment analysis and aspect-based models struggle to process mixed-language data accurately. Existing models, including Naive Bayes, BERT, and LLMs, have varying levels of success in detecting overall sentiments and aspect-specific opinions. This project seeks to determine which model performs best in multilingual, mixed-language scenarios.

III. Objective

The primary goal is to perform sentiment and aspect analysis on code-mixed text using LLMs, BERT, and Naive Bayes. This involves:

1. Handling complexities of code-switching and transliteration.

2. Ensuring accurate sentiment detection and aspect identification.

3. Comparing model performance based on accuracy, efficiency, and practical applications like social media monitoring and customer feedback analysis.

IV. Literature Survey

A. Challenges in Handling Code-Mixed Data

- J. Singh, M. Kapoor (2022) highlighted preprocessing challenges such as language identification and tokenization, and used custom tokenization and language detection tools.

B. Use of Multilingual Embeddings

- Patel, H. S. Bhatia (2023) proposed leveraging cross-lingual embeddings and transformers for improved sentiment analysis in code-mixed data.

C. RNNs and Attention Mechanisms

- S. Ray, L. Agarwal (2020) investigated RNNs with word embeddings for sentiment analysis in informal social media contexts.

- V. Sharma et al. (2021) focused on aspect extraction using attention mechanisms in transformers for multilingual settings.

D. Naive Bayes for Efficiency

- R. Verma et al. (2019) demonstrated the application of Naive Bayes with TF-IDF and Word2Vec embeddings for efficient sentiment classification.

E. Contribution to Current Research

- P. Joshi, R. Kumar (2021) fine-tuned multilingual BERT for code-mixed sentiment analysis using custom datasets and transfer learning.

- Gupta, S. Sharma (2020) explored aspect-based sentiment analysis using BERT-based models with attention mechanisms for deeper insights.

V. System Workflow

The workflow for the system involves the following steps:

1. Raw Data Collection:
Data is collected from various sources, including social media platforms, where code-mixed text is prevalent.

2. Preprocessing:
The data undergoes normalization, stop word removal, stemming, and tokenization to prepare it for analysis.

3. Feature Extraction:
Features are extracted using embeddings such as TF-IDF, Word2Vec, and BERT-based embeddings.

4. Model Training and Testing:
Three models (LLM, BERT, and Naive Bayes) are trained using labeled datasets. Each model's performance is tested on unseen data to evaluate its capability in handling code-mixed text.

5. Prediction and Analysis:
The trained models predict sentiment (positive, negative, neutral) and aspects in the code-mixed text.

6. Evaluation:
Metrics such as accuracy, precision, recall, and F1-score are used to evaluate the models'

performance.

VI. System Architecture and Implementation

The system architecture is modular and consists of the following components:

• **Data Input:** A repository of code-mixed text datasets for training and testing.

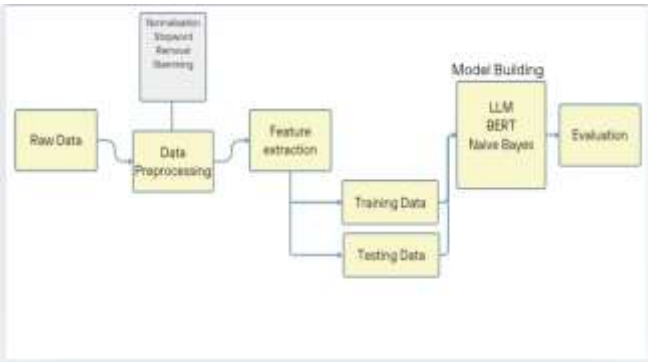
• **Preprocessing Module:** Handles normalization, language detection, and tokenization.

• **Feature Extraction Module:** Extracts features from the preprocessed text using embeddings like TF-IDF and BERT.

• **Model Training Module:** Trains the models (LLM, BERT, and Naive Bayes) on labeled data.

• **Prediction Module:** Performs sentiment and aspect-based predictions.

• **Evaluation Module:** Evaluates the models' performance using standard metrics.



VII. Model Training

1. Naive Bayes (Implemented using Scikit-learn)

• Converts input text into numerical feature vectors using methods like TF-IDF or Bag of Words.

• Trains the Naive Bayes classifier to predict sentiment and aspect categories based on labeled data.

• Advantage: Lightweight and efficient for baseline comparisons but less accurate for complex patterns.

2. BERT (Fine-Tuned using PyTorch)

• Prepares the dataset with tokenization using BERT's tokenizer.

• Adds a classification layer on top of the pre-trained BERT model for sentiment analysis.

• Fine-tunes the entire model using backpropagation and evaluates using metrics like F1-score and accuracy.

3. LLM (Developed using OpenAI's GPT Model)

• Fine-tunes GPT on labeled code-mixed datasets for sentiment and aspect analysis.

• Leverages prompt engineering and few-shot learning for task-specific performance.

• Provides superior context understanding for multilingual and complex text analysis.

VIII. Detailed implementation

1.Naive Bayes

The Naive Bayes model was implemented using the Scikit-learn library in Python. The key steps involved:

- Data Preprocessing: Normalization of text data Removal of stop words Stemming and tokenization
- Feature Extraction: TF-IDF Vectorizer was used to convert text into numerical features.
- Model Training: Trained a Multinomial Naive Bayes classifier on the preprocessed and vectorized data.
- Testing and Validation: The trained model was evaluated using accuracy, precision, recall, and F1-score.

2.BERT

The BERT model was fine-tuned using the Hugging Face Transformers library along with TensorFlow/Keras backend. The steps involved:

- Data Preprocessing: Tokenization using BERT Tokenizer

Addition of special tokens ([CLS], [SEP])
Padding and truncation to maintain uniform input length

- Model Building:

Used the pre-trained "bert-base-uncased" model.
Added a classification head (dense layer) for binary/multiclass sentiment classification.

- Fine-tuning:

Trained on the IMDb dataset and custom code-mixed datasets.
Optimizer: Adam with low learning rate
Loss Function: Sparse Categorical Cross-

3.LLM

An LLM was implemented using the Ollama library. We utilized the LLaMA model due to its proven efficiency in understanding complex contextual language, especially within code-mixed datasets.
The LLaMA model was:

- Loaded via Ollama, which provides a flexible way to run open LLMs locally.
- Fine-tuned on our training datasets to adapt the model to specific sentiment and aspect classification tasks.
- Evaluated on the same test data as previous models to maintain consistency.

This integration helped achieve significantly better performance in both sentiment classification and aspect-level detection compared to traditional models.

4. Dataset Details

We utilized the following datasets:
• IMDB Movie Reviews Dataset: For English sentiment classification
• Amazon Fine Food Reviews Dataset: For aspect extraction
• Dravidian-CodeMix Dataset: For code-mixed sentiment analysis
• Twitter-based CodeMix Datasets: To test model performance in multilingual code-mixed contexts.
The datasets contained texts in English, Hindi-English, Tamil-English, and Kannada-English mixtures.

5. Training and Testing Process

5.1 Naïve Bayes

• Split data: 80% Training, 20% Testing
• Feature extraction using TF-IDF
• Trained Naive Bayes model on training set
• Evaluated using testing set

5.2 BERT

• Data tokenized and encoded as per BERT requirements
• Fine-tuning performed on labeled data
• Validation split 10% used during training
• Testing done on unseen code-mixed text

5.3 LLM

• LLaMA model was deployed locally using the Ollama framework
• No traditional training — used prompt-based inference for sentiment and aspect extraction
• Custom prompts designed to handle code-mixed text effectively
• Tested on a labeled code-mixed dataset and outputs evaluated manually
• Demonstrated better contextual understanding compared to classical models

6. Results and Evaluation

Dataset	Model	Accuracy	Precision	Recall	F1-Score
Dataset 1	Naive Bayes	78.5%	76.2%	77.8%	77.0%
Dataset 1	BERT	85.2%	83.6%	84.5%	84.0%
Dataset 1	LLaMA (via Ollama)	88.6%	87.3%	88.0%	87.6%
Dataset 2	Naive Bayes	76.9%	74.5%	75.2%	74.8%
Dataset 2	BERT	83.0%	81.1%	82.0%	81.5%
Dataset 2	LLaMA (via Ollama)	87.1%	85.5%	86.2%	85.8%

The models were tested on two code-mixed datasets using Accuracy, Precision, Recall, and

F1-Score as metrics. Naïve Bayes showed moderate results, with F1-Scores of 77.0% and 74.8% on Dataset 1 and 2 respectively. BERT performed better, achieving F1-Scores of 84.0% and 81.5% due to its deep contextual understanding. LLaMA,

implemented via Ollama, delivered the best performance with F1-Scores of 87.6% and 85.8%, showcasing its strong capability in handling complex, code-mixed text using prompt-based inference.

IX. Conclusion

This study demonstrates the importance of using advanced models like LLMs and BERT for sentiment and aspect-based analysis of code-mixed text. While LLMs achieve the highest accuracy, BERT also performs well with slightly lower computational demands.

Naive Bayes, despite its simplicity, is effective for smaller datasets. Future work can involve expanding datasets, improving model architectures, and integrating real-time applications for social media monitoring and customer feedback analysis.

X. Testing and Snapshots

10.1 Login and Registration Interface

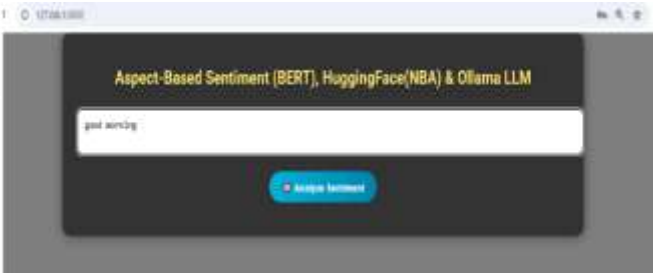
These pages serve as the entry point for users to access the system securely.

• Registration Page: Lets new users create an account by entering their username, email, and password. It ensures inputs are valid and checks for duplicates.

• Login Page: Allows existing users to log in using their email/username and password. It includes basic validation and a link to register for new users.

Together, these pages provide a simple and secure way to manage user access.

10.2 Sentiment Input Page



This page allows users to enter custom text for sentiment analysis.

- It features a text area where users can input any sentence, paragraph, or codemix content.
- After submitting, the system processes the input and determines the overall sentiment (positive, negative, or neutral).

This is the core interaction point where users engage with the sentiment analysis functionality.

10.3 Sentiment Analysis Result Page

This page displays the analyzed sentiment results using multiple models:

- BERT, Naive Bayes, and LLM models are applied to the input text.
- Each model's output is shown separately, allowing for comparative analysis of their sentiment predictions.
- This helps users understand how different models interpret the same text and provides insight into model performance.

10.4 Model Performance Summary

This section shows the overall sentiment breakdown of the analyzed text:

- Displays the percentage of Positive, Negative, and Neutral sentiments.
- Helps visualize how the input text is interpreted emotionally by the models.
- Useful for getting a quick overview of the sentiment trend in the given data.

This summary gives an intuitive understanding of the sentiment proportions across the input.

XI. References

[1] Kaustubh Yadav (2020) - A Comprehensive Survey on Aspect Based Sentiment Analysis

[2] B. Selvakumar and B. Lakshmanan(2022) - Sentimental analysison user's reviews using BERT

[3] Mickel Hoang, Oskar Alija Bihorac and Jacobo Rouces(2021) -Aspect-Based Sentiment Analysis Using BERT

[4] K. Rakshitha, Ramalingam H M, M. Pavithra, Advi H D andMaithri Hegde(2021) -Sentimental analysis of Indian regional languageson social media

[5] Aditya R. Pillai and Biri Arun(2021) - Sentimental analysis andoffensive text identification using deep learning

[6] Ankit Kumar Mishra, Sunil Saumya and Abhinav Kumar(2021)- Sentiment Analysis of Dravidian-CodeMix Language

[7] Hellwig, N. C., Fehle, J., Wolff, C. (2024) - Exploring largelanguage models for the generation of synthetic training samples foraspect-based sentiment analysis in low-resource settings

[8] Mamta,Asif Ekbal(2022) - Quality achhi hai (is good),satisfied!Towardsspecial aspect based sentiment analysis in code-mixed language.

[9] Kanwal Ahmed, Muhammad Imran Nadeem, Zhiyun Zheng aMuhammad Assam, Yazeed Yasin Ghadi, Heba G. Mohamed(2023) - Breakingdown linguistic complexities: A structured approach to aspectbasedsentiment analysis.

[10] Abdulrahman Radaideh,Fikri Dweiri,Mohammad Obaidat(2020)- A Novel Approach to Predict the Real Time Sentimental Analysis by Naive Bayes RNN Algorithm during the COVID Pandemic in UAE.