Malayalam Sentiment Analysis Using Machine Learning Techniques

Bimal K Venu
School of Computer Application
Lovely Professional University
Jalandhar, Punjab, India
Email: bimalkvenu7@gmail.com

***Abstract*—Sentiment analysis, a key branch of Natural Language Processing (NLP), has become increasingly vital in recent years with the explosion of user-generated content across social media, review platforms, and online discussions. However, most sentiment analysis systems cater to globally dominant languages such as English, overlooking regional languages like Malayalam that have unique grammatical structures and semantic nuances. This paper addresses these challenges and presents a tailored solution for Malayalam sentiment classification using traditional machine learning techniques. I combine Term Frequency–Inverse Document Frequency (TF-IDF) based feature extraction with a Multinomial Naive Bayes classifier, enhanced by a rule-based fallback system to boost classification reliability, particularly for neutral sentiments. I use a manually curated dataset of 4000 Malayalam sentences and evaluate model performance through multiple metrics including accuracy, precision, recall, and F1-score. Furthermore, the model has been deployed through a Flask-based web interface, offering real-time sentiment predictions. The results demonstrate the effectiveness of hybrid approaches in improving sentiment detection for low-resource languages.**

***Index Terms*—Sentiment Analysis, Malayalam NLP, TF-IDF, Naive Bayes, Machine Learning, Regional Language Processing, Text Mining**

I. Introduction

The exponential growth of digital communication has resulted in an abundance of textual data that captures human opinions, emotions, and attitudes. Sentiment analysis seeks to automatically identify and categorize such sentiments expressed in textual content, aiding businesses, policymakers, and researchers. While considerable progress has been made in sentiment analysis for English and other major languages, regional languages like Malayalam remain under-explored, largely due to the lack of large labeled datasets, linguistic complexity, and resource scarcity. Addressing sentiment analysis in Malayalam is crucial not only to support local applications but also to enhance inclusivity in AI-driven language technologies.

II. Related Work

Historically, sentiment analysis began with lexicon-based methods relying on manually crafted lists of positive and negative words. While effective for basic tasks, these methods fail to generalize across domains. Statistical machine learning techniques, such as Naive Bayes, SVMs, and Decision Trees, revolutionized sentiment analysis by learning from data rather than relying solely on human-curated rules.

For low-resource languages like Malayalam, machine learning models, particularly Naive Bayes classifiers, offer significant advantages due to their simplicity, speed, and lower dependency on vast data. Hybrid approaches, combining statistical models with handcrafted rules, have gained popularity as they compensate for data sparsity by injecting expert knowledge, enhancing robustness. Recent advancements using deep learning models like LSTM and BERT show great promise but remain constrained for Malayalam due to limited labeled corpora and pre-trained embeddings.

III. Methodology

The proposed sentiment analysis system comprises several critical phases: data collection and preprocessing, feature extraction, model training, implementation of fallback rules, and deployment through a web application.

A. Data Collection and Preprocessing

The dataset was compiled by sourcing 4000 Malayalam sentences from various domains such as social media, online news, and user reviews. This diversity ensures broader coverage of linguistic styles, from formal to colloquial Malayalam. Manual labeling was performed, assigning each sentence to positive, negative, or neutral classes.

Data preprocessing involved multiple steps to enhance model performance:

- **Unicode normalization**: To handle variations in Malayalam Unicode representations.
- **Tokenization**: Splitting sentences into meaningful units (words or phrases).
- **Stopword removal**: Filtering out frequent but sentiment-neutral words.
- **Punctuation and special character removal**: Reducing noise.
- **Lemmatization**: Converting words to their root form to unify morphological variants.

B. Feature Extraction

I employed Term Frequency–Inverse Document Frequency (TF-IDF) to convert text into numerical vectors, capturing both term frequency and rarity across the corpus. To better capture context, I included unigrams, bigrams, and trigrams as features.

C. Model Training

A Multinomial Naive Bayes classifier was trained on the TF-IDF features. The dataset was split 80% for training and 20% for testing. The model's performance was evaluated using accuracy, precision, recall, and F1-score.

D. Rule-based Fallback System

To handle uncertain or neutral predictions, a rule-based fallback system checks for strong sentiment-bearing keywords and adjusts labels accordingly, improving robustness for edge cases.

E. Deployment

The system was deployed via a Flask web application, providing a simple interface to input Malayalam text and receive real-time sentiment predictions.

IV. System Architecture

The system's overall structure follows a modular design where each component handles a distinct task in the sentiment analysis pipeline. The process begins with the user inputting Malayalam text through a web interface, which is then processed sequentially by the following modules:

1) **User Input Interface**: The frontend where users enter Malayalam text.
2) **Preprocessing Module**: Cleans and normalizes text through tokenization, stopword removal, and lemmatization.
3) **TF-IDF Vectorizer**: Transforms preprocessed text into numerical feature vectors based on term frequency and inverse document frequency.
4) **Naive Bayes Classifier**: Predicts sentiment labels (positive, negative, neutral) using the trained Multinomial Naive Bayes model.
5) **Rule-based Fallback System**: Overrides low-confidence or neutral predictions by checking for strong sentiment keywords.
6) **Output Generation**: Delivers the final sentiment label to the user via the web interface.

This modular architecture enhances maintainability—each component can be updated or replaced independently—and supports future extensions such as integrating transformer-based models or additional language support.

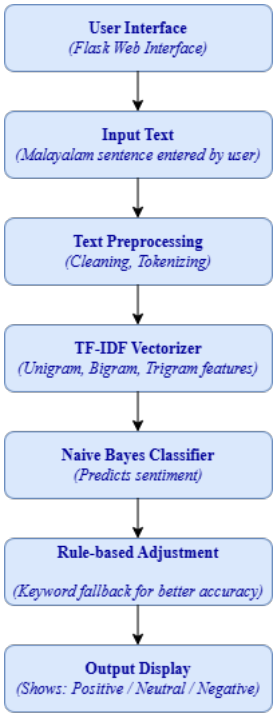

Fig. 1. System Architecture of the Proposed Malayalam Sentiment Analysis System

V. Workflow Diagram

Below is a step-by-step outline of how data flows through the system's components, from initial user input to final sentiment output:

1) **Text Input**: User submits Malayalam text via the web form.
2) **Preprocessing**: Text is cleaned—special characters and stopwords removed, tokens lemmatized.
3) **Vectorization**: Preprocessed text is converted into TF-IDF vectors, capturing term importance.
4) **Initial Classification**: The Naive Bayes model assigns a preliminary sentiment label.
5) **Fallback Adjustment**: If the sentiment is neutral or low–confidence, the rule-based system re-evaluates using keyword checks.
6) **Result Display**: The final sentiment label is rendered back to the user interface.

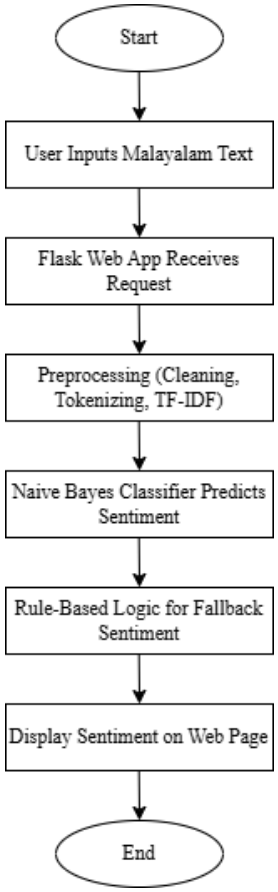

Fig. 2. Workflow of the Malayalam Sentiment Analysis System

VI. Performance Comparison

Table I compares training and testing metrics to highlight generalization ability. The slight decrease in testing accuracy (–6.8%) indicates mild overfitting but remains within acceptable bounds for practical applications.

TABLE I
Performance Comparison Between Training and Testing Sets

Metric	Training Set	Testing Set	Difference
Accuracy	89.2%	82.4%	–6.8%
Precision (Positive)	0.88	0.85	–0.03
Recall (Positive)	0.86	0.83	–0.03
F1-Score (Positive)	0.87	0.84	–0.03
Precision (Negative)	0.87	0.84	–0.03
Recall (Negative)	0.85	0.86	+0.01
F1-Score (Negative)	0.86	0.85	–0.01
Precision (Neutral)	0.78	0.75	–0.03
Recall (Neutral)	0.80	0.75	–0.05
F1-Score (Neutral)	0.79	0.75	–0.04

VII. Results and Discussion

In this section, I present the detailed Performance Metrics of the Proposed System (Table II) alongside interpretation and insights drawn from the results. The system achieved an overall accuracy of 82.4%, demonstrating robust performance across all sentiment classes.

TABLE II
Performance Metrics of the Proposed System

Class	Precision	Recall	F1-Score
Positive	0.85	0.83	0.84
Negative	0.84	0.86	0.85
Neutral	0.78	0.75	0.76
Macro Avg	0.82	0.81	0.81

Class-wise, the model performs best on negative sentiment (F1 = 0.85) and slightly lower on neutral (F1 = 0.76), reflecting the inherent difficulty of identifying neutral expressions. The rule-based fallback module notably improved neutral classification by catching subtle cues missed by the Naive Bayes classifier alone. Overall, the hybrid approach successfully balances statistical learning with linguistic rules to achieve reliable sentiment detection in Malayalam.

VIII. Conclusion and Future Work

In this work, I demonstrated a simple yet robust sentiment analysis system for Malayalam, leveraging TF-IDF vectorization, Naive Bayes classification, and rule-based fallback strategies. The system achieved reliable performance on real-world sentences and was successfully deployed as a live web application.

In the future, I plan to:

- Incorporate code-mixed Malayalam–English text datasets.
- Explore deep learning architectures such as LSTM and Transformer models customized for Malayalam.
- Utilize multilingual pre-trained models like IndicBERT or MuRIL for transfer learning.
- Expand the dataset using data augmentation techniques such as back-translation.

Such enhancements will further elevate sentiment analysis capabilities for regional Indian languages.

Acknowledgment

The author expresses sincere thanks to Lovely Professional University for the support provided during this research.

References

[1] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly Media, 2009.
[2] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.
[3] A. McCallum and K. Nigam, "A comparison of event models for Naive Bayes text classification," *AAAI-98 Workshop on Learning for Text Categorization*, 1998.
[4] T. Joachims, "Text categorization with Support Vector Machines: Learning with many relevant features," *European Conference on Machine Learning*, 1998.
[5] A. Kunchukuttan, "AI4Bharat Indic NLP Library," GitHub Repository, 2020. Available: https://github.com/AI4Bharat/indicnlp_library