Indian Journal of Computer Science and Technology
Volume 5, Issue2 (May-August 2026), PP: 761-767.
www.indjcst.com **ISSN No: 2583-5300**

Tai Ahom Sentiment Analysis System Using Lexicon-Based and Naive Bayes Approaches

Racktutpal Khataniar[1], Dr. Dhrubajyoti Baruah[2]
[1,2]Department of Computer Application, Jorhat Engineering College, Assam, India.

To Cite this Article: *Racktutpal Khataniar[1], Dr. Dhrubajyoti Baruah[2], "Tai Ahom Sentiment Analysis System Using Lexicon-Based and Naive Bayes Approaches", Indian Journal of Computer Science and Technology, Volume 05, Issue 02 (May-August 2026), PP: 761-767.*

Abstract: *This work presents a sentiment analysis system for the Tai Ahom language, addressing the scarcity of Natural Language Processing resources for ancient, low-resource languages of Northeast India. Tai Ahom was the court and literary language of the Ahom kingdom that ruled Assam for nearly six hundred years, and its script is now part of the Unicode Standard, yet no sentiment lexicon, labelled dataset, or analysis pipeline previously existed for it. The proposed system classifies Tai Ahom text directly, in both native script and romanized form, into positive, negative, and neutral sentiment categories, without using any translation step. The framework combines a manually constructed, multi-rater sentiment lexicon of 498 words with a Complement Naive Bayes classifier from scikit-learn, applied directly to native Tai Ahom text. A hybrid sequential pipeline combines both approaches using a confidence-based fallback strategy. The application is implemented using Streamlit for an interactive interface and Ngrok for secure web-based deployment. Experimental evaluation on 392 labelled test sentences shows that the Naive Bayes model achieves 92.1% accuracy, considerably outperforming the standalone lexicon model (52.8%) and the hybrid pipeline (64.5%). These results are compared against AABEG, a related study on Assamese sentiment analysis that used Google Translate followed by VADER and Naive Bayes, where the translation-based VADER approach scored 0% accuracy. This comparison confirms that working directly with native language text gives far better results than translating it first.*

Keywords: *Sentiment Analysis; Tai Ahom Language; Natural Language Processing (NLP); Low-Resource Languages; Machine Learning; Lexicon-Based Model; Naive Bayes.*

I. INTRODUCTION

The Tai Ahom Sentiment Analyzer project is developed to create an intelligent system capable of identifying and classifying sentiments expressed in Tai Ahom language text. Tai Ahom, the historical language of the Ahom kingdom that ruled Assam from 1228 CE to 1826 CE, is recognized as a severely low-resource language in Natural Language Processing due to the scarcity of annotated datasets, linguistic corpora, and computational tools. This study aims to address this gap by building a system that categorizes Tai Ahom text into positive, negative, or neutral sentiments using two distinct approaches, a lexicon-based model and a Naive Bayes classifier, implemented and evaluated both separately and together.

The first approach employs a manually constructed sentiment lexicon, similar in spirit to VADER (Valence Aware Dictionary and sEntiment Reasoner), a lexicon and rule-based sentiment analysis technique known for its effectiveness on informal text. The lexicon assigns sentiment scores to Tai Ahom words and phrases, with negation and intensifier handling rules applied during scoring to allow for nuanced sentiment evaluation directly in the native language.

The second approach utilizes the Naive Bayes classifier, a probabilistic machine learning algorithm that learns sentiment categories from labelled datasets. Naive Bayes is valued for its simplicity, efficiency, and robustness in text classification tasks, particularly when training data is limited or moderately sized, which makes it well suited to a newly built dataset such as this one.

Unlike prior work on the related Assamese language, where the Google Translator API was used to translate text into English before applying existing English-language NLP tools, this system is designed from the outset to avoid translation entirely. The system is implemented in Python, leveraging scikit-learn for machine learning and pandas for data processing. A user-friendly web interface is developed using Streamlit, which allows users to input Tai Ahom text, in either native script or romanized form, and instantly view sentiment predictions from all models. Ngrok is used to provide secure, remote access to the deployed system.

The two techniques, lexicon-based matching and Naive Bayes, were tested independently and then combined into a hybrid pipeline, with their outputs compared in terms of accuracy and class-wise reliability. The results indicate that the lexicon model performs adequately for high-confidence native phrases, while Naive Bayes achieves considerably higher overall accuracy once trained on the available labelled data.

This project demonstrates the feasibility of performing sentiment analysis for a severely low-resource ancient language entirely in its native form. Its potential applications include digital preservation of historical manuscripts, educational tools for Tai Ahom script learning, and research support for computational linguistics in Northeast India.

II.LITERATURE REVIEW

International:

The study of sentiment analysis has developed step by step over the years. Early sentiment work in the late 1990s and early 2000s focused on how computers could understand opinions expressed through adjectives and word choice. As research progressed

through the 2000s, attention shifted toward building sentiment dictionaries and rule-based systems capable of handling negation, intensifiers, and contextual polarity shifts.

With the growth of informal digital text, Hutto and Gilbert (2014) developed VADER, a widely used rule-based model for sentiment analysis of social media text. It uses a dictionary of words with associated sentiment scores and handles punctuation, negation, and emphasis effectively. From the mid-2010s onward, machine learning methods became dominant for text classification, with Naive Bayes and similar probabilistic classifiers widely adopted due to their strong accuracy and ease of implementation, even on comparatively small datasets. More recently, multilingual and cross-lingual research has grown substantially, as researchers aim to extend sentiment analysis techniques to low-resource languages around the world.

National:

Sentiment analysis research in Indian languages has developed gradually, though challenges such as linguistic complexity, lack of annotated datasets, and limited digital resources continue to slow progress. Studies in major Indian languages including Bengali, Hindi, and Telugu have applied both lexicon-based and machine learning approaches, often relying on translation to English to overcome the scarcity of native sentiment tools.

For Assamese, a language closely related to the cultural context of Tai Ahom, research is still in comparatively early stages. The most directly relevant prior work is AABEG (Analysis of Assamese Backed English Generated Sentiment) by Baruah and Boruah, which translated Assamese text into English using the Google Translate API before applying VADER and Naive Bayes classifiers. Their VADER-based approach achieved 0% accuracy on the test set, while their Naive Bayes model improved substantially with additional training data, ultimately reaching 85% accuracy. This result highlights both the risk of translation-based pipelines for low-resource languages and the continued reliability of Naive Bayes as a machine learning baseline even with comparatively limited data.

For Tai Ahom specifically, no prior sentiment analysis system, lexicon, or labelled dataset existed before this work. This gap highlights the importance of developing native, translation-free sentiment analysis tools and datasets specifically for ancient and severely low-resource languages such as Tai Ahom.

III. WORKING MECHANISM

Lexicon Model Mechanism:

The lexicon model's core mechanism operates in several stages. First, the input text is tokenized, and each token is matched against a sentiment lexicon of 498 Tai Ahom words and phrases, built from native script and romanized vocabulary sources. Unlike supervised machine learning models that require large labelled datasets, the lexicon model relies on a pre-built sentiment dictionary and a small set of grammatical rules to determine sentiment polarity directly. The steps are summarized below.

Step	Process	Description / Function	Example
1	Tokenization	Text is split into individual tokens (words).	"ni khen koi" → [ni, khen, koi]
2	Lexicon Lookup	Each token or phrase is matched against lexicon entries with averaged multi-rater scores.	"ni" → +1.5; "khen" → +2.0
3	N-gram Matching	Longest-match strategy tries 3-word, then 2-word, then 1-word phrases.	"ngai khen di" matched as one phrase
4	Negation and Intensifier Rules	Negation words reverse polarity; intensifier words scale it upward.	"bau ni" → polarity reversed
5	Score Aggregation	Combines all matched word-level scores into one sentence score.	Average of all matched scores
6	Threshold Classification	Final score is classified as Positive, Negative, or Neutral.	Score ≥ 0.3 → Positive; ≤ -0.3 → Negative

Table I. Lexicon Model Processing Steps

The multi-rater lexicon score for each word is computed as the arithmetic mean of all available rater columns:

$$S(w) = (1 / n) \times \Sigma_{i=1}^{n} r_i(w)$$

where $S(w)$ is the final merged sentiment score for word w, n is the number of raters who scored that word, and $r_i(w)$ is the score given by rater i. This averaging reduces individual annotator bias, particularly important since Tai Ahom has no existing reference lexicon to validate against.

Naive Bayes Mechanism:

The Naive Bayes classifier is a probabilistic machine learning model based on Bayes' theorem, commonly used for text classification tasks such as sentiment analysis. It assumes that all features, in this case words or tokens, are conditionally independent given the class label. Despite this simplifying assumption, the model performs remarkably well in practice, especially for moderately sized text datasets.

The foundation of Naive Bayes is Bayes' theorem, expressed mathematically as:

$$P(C|X) = [P(X|C) \times P(C)] / P(X)$$

Where P(C|X) is the probability of a sentiment class C (positive, negative, or neutral) given the observed words X in a sentence, P(X|C) is the likelihood of observing those words given the class, P(C) is the prior probability of the class, and P(X) is the overall probability of the observed words, acting as a normalizing constant.

Because the Tai Ahom test dataset has an uneven distribution across sentiment classes, with neutral and positive sentences considerably outnumbering negative ones, a Complement Naive Bayes (CNB) classifier was used in place of standard Naive Bayes. Complement Naive Bayes estimates each class's parameters using data from all other classes rather than the class itself, which corrects for the bias that standard Naive Bayes exhibits toward majority classes:

$$\theta^{c}_{i} = (\alpha_i + \Sigma\square:_{\gamma}\square\neq^{c} d\square_i) / \tilde{\Sigma}_i (\tilde{\alpha_i} + \Sigma\square:_{\gamma}\square\neq^{c} d\tilde{\square_i})$$

Where θ^{c}_{i} is the estimated weight of feature i for class c, $d\square_i$ is the count of feature i in document j, $y\square$ is the true label of document j, and α_i is a smoothing parameter. This formulation prevents the dominant sentiment classes from overwhelming the scarcer Negative class during training.

Step	What Happens	Example (Sentiment)
1	Collect training data with labels	"ni khen koi" → Positive, "bau ni" → Negative
2	Convert text into TF-IDF feature vectors	Words and word-pairs weighted by importance
3	Count word frequency in each class	"khen" appears often in Positive, rarely in Negative
4	Find the overall chance of each class	37.9% Positive, 12.0% Negative, 50.3% Neutral
5	For a new sentence, check likelihood per class	Compute probability for Positive, Negative, Neutral
6	Predict the class with the highest probability	Positive score > others → Predict Positive

Table II. Naive Bayes Processing Steps

Hybrid Sequential Fallback Mechanism:

The hybrid model combines both mechanisms using aconfidence-based decision rule. The final sentiment label is assigned using:

$$\text{Label}(S) = \text{Positive, if } S \geq 0.3 \quad \text{Label}(S) = \text{Negative, if } S \leq -0.3 \quad \text{Label}(S) = \text{Neutral, otherwise}$$

If the lexicon model is confident, its result is used directly. If the lexicon finds no matching words at all, the Naive Bayes prediction is used as a fallback. In the remaining cases, the two models' scores are blended together, weighted more heavily toward the lexicon since it reflects carefully checked human judgement.

IV. FRAMEWORK DEVELOPMENT

Streamlit is an open-source Python framework designed to create data-driven web applications with minimal effort. It is especially popular among data scientists and machine learning engineers who want to build interactive tools and dashboards without needing extensive web development experience. It allows developers to build dynamic interfaces using familiar Python syntax, integrating seamlessly with libraries such as pandas and Plotly, while handling underlying web development complexities such as HTML, CSS, and JavaScript internally.

scikit-learn is a comprehensive Python library used for machine learning tasks in this project, providing the TfidfVectorizer for feature extraction and the ComplementNB classifier used for sentiment classification. It serves as a robust and well-tested framework for building and evaluating text classification models.

Ngrok is a tool that allows a local development server to be securely exposed to the internet. In simple terms, it lets a locally running web application, such as one built with Streamlit on localhost, be accessed from anywhere using a public URL, without configuring firewalls or deploying to external hosting. This made it possible to demonstrate the working system remotely to the project guide during development.

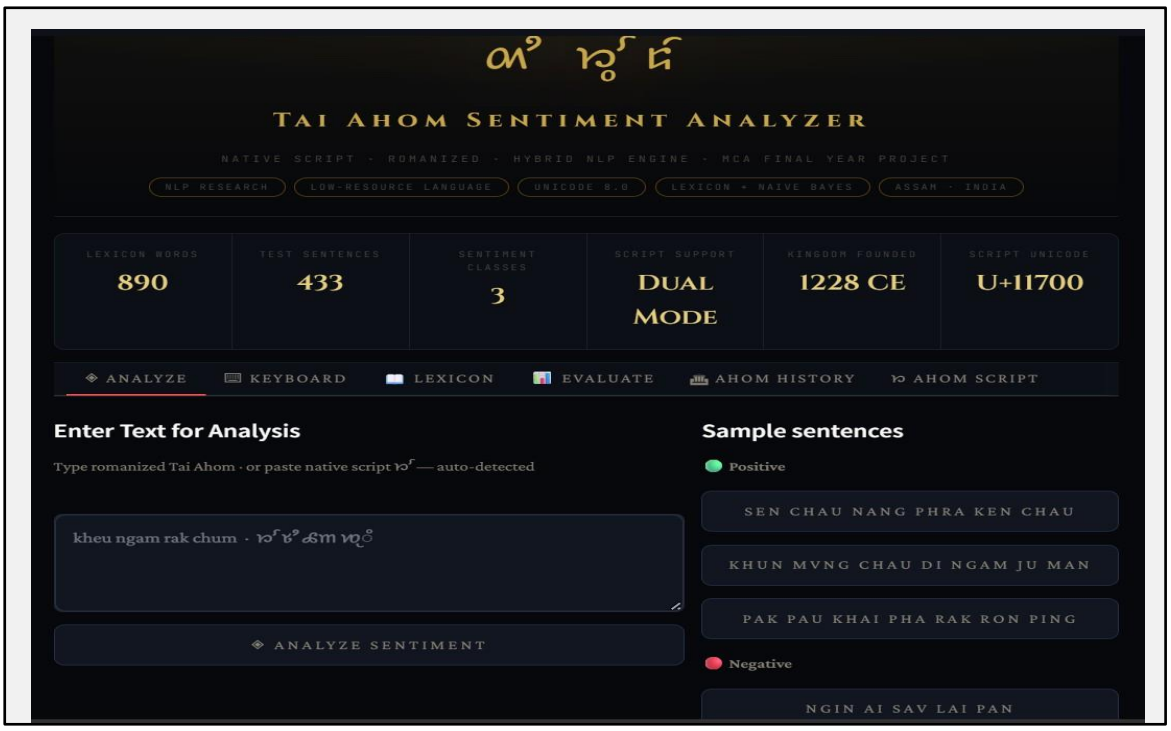

Figure 1 Screenshot of the Streamlit application home interface showing the project overview and dataset summary panel.

The lexicon loading and merging logic automatically detects all available rating columns in the source CSV files and averages them at run time, so that adding additional rater columns in the future requires no code changes. Duplicate word entries across source rows are grouped and their scores combined rather than allowed to silently overwrite one another, ensuring that every rater's input is correctly reflected in the final lexicon

Figure 2 Screenshot of the Analyze tab showing the native Tai Ahom virtual keyboard alongside the text input area and sentiment results from all models.

.

V.DATA AND RESULT

Test data was chosen and evaluated across all three models: the standalone lexicon model, the standalone Naive Bayes model, and the hybrid sequential pipeline.

Dataset Component	Count	Notes
Lexicon2_0.csv (multi-rater)	438 words	4 rating columns averaged
Lexicon.csv (secondary)	354 words	Supplementary coverage
Merged unique lexicon	498 words	Deduplicated, native + romanized
Test sentences (positive)	164	37.9% of test set
Test sentences (negative)	52	12.0% of test set
Test sentences (neutral)	218	50.3% of test set
Total unique test sentences	392	Deduplicated from 433

Table III. Dataset Distribution

Here, it is noted that the standalone lexicon model shows weaker performance on the negative class specifically, primarily due to limited negative-sentiment vocabulary coverage in the current lexicon. There is clear scope to improve this by expanding negative-sentiment word entries. For Naive Bayes, performance can also be improved further by adding more labelled training data, similar to how AABEG's Naive Bayes model improved substantially when trained with more data.

Model	Accuracy	Macro F1	Precision	Recall
Lexicon (Standalone)	52.8%	0.454	0.462	0.463
Naive Bayes (Standalone)	92.1%	0.900	0.882	0.921
Hybrid Sequential Pipeline	64.5%	0.599	0.623	0.645

Figure 3
Screenshot of the Evaluate tab showing the accuracy bar chart comparison across all three models.

Table IV. Overall Model Performance Comparison

Confusion Matrices:

The confusion matrices below show actual sentiment labels as rows and predicted labels as columns for each model, evaluated on the 392-sentence test set.

Actual \ Predicted	Positive	Negative	Neutral
Positive	86	25	27
Negative	17	11	17
Neutral	56	43	110

Table V. Confusion Matrix — Lexicon Model

Actual \ Predicted	Positive	Negative	Neutral
Positive	127	7	4

Negative	2	42	1
Neutral	11	6	192

Table VI. Confusion Matrix — Naive Bayes Model

Actual \ Predicted	Positive	Negative	Neutral
Positive	117	17	4
Negative	12	27	6
Neutral	57	43	109

Figure 4 Screenshot of the Evaluate tab showing all three confusion matrices rendered as heatmaps, side by side.

Table VII. Confusion Matrix — Hybrid Pipeline

So it shows that the standalone Naive Bayes model performs considerably better in comparison to both the lexicon model and the hybrid pipeline. Many test cases that contained vocabulary not present in the lexicon were correctly classified by Naive Bayes, since it learns statistical word-sentiment associations directly from the training data rather than depending on a fixed dictionary.

Comparing these results with AABEG: their VADER approach scored 0% accuracy, their less-trained Naive Bayes scored 45%, and their more-trained Naive Bayes scored 85%, all using translated Assamese text. The Naive Bayes model in this study, working entirely on native Tai Ahom text with no translation, reached 92.1% accuracy, exceeding even AABEG's best result.

System / Model	Accuracy	Notes
AABEG - VADER (translated)	0.00%	Google Translate then VADER
AABEG - Naive Bayes (less data)	45.0%	Translated text, smaller dataset
AABEG - Naive Bayes (more data)	85.0%	Translated text, larger dataset
Ours - Naive Bayes	92.1%	Native Tai Ahom, no translation

Table VIII. Comparison With Aabeg (Translation-Based)

VI.CONCLUSION

The findings of this study highlight the superior performance of the machine learning based Naive Bayes classifier compared to the lexicon-based approach, demonstrating that models trained on relevant native-language data can achieve highly accurate sentiment classification results for Tai Ahom. Furthermore, by avoiding language translation entirely, the results confirm that native script and native spelling sentiment analysis is not just a viable approach but a clearly superior one for processing severely low-resource ancient languages, opening avenues for future research in Tai-Kadai language processing and regional heritage language preservation.

The corresponding findings provide a replicable framework for developing sentiment analysis systems in other low-resource and ancient languages, demonstrating that satisfactory performance can be attained even in the absence of extensive linguistic or computational resources. The developed system enables researchers, educators, and cultural organisations to apply sentiment analysis to Tai Ahom manuscripts and texts, contributing to greater digital inclusivity for endangered language communities. The limitations and error patterns identified in this study, particularly the lexicon's current weakness on negative-sentiment vocabulary, offer clear directions for future research, including the expansion of the Tai Ahom sentiment lexicon, the adoption of native-script machine learning architectures, and the incorporation of contextual and domain-specific understanding to further improve performance.

REFERENCES

1. Baruah, D., & Boruah, A. (2025). Analysis of Assamese Backed English Generated Sentiment: AABEG. Indian Journal of Computer Science and Technology, 4(3), 203–211.
2. Hutto, C., & Gilbert, E. (2014). VADER: A parsimonious rule-based model for sentiment analysis of social media text. Proceedings of the 8th International AAAI Conference on Weblogs and Social Media (ICWSM).
3. Rennie, J. D., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the poor assumptions of Naive Bayes text classifiers. Proceedings of the

20th International Conference on Machine Learning (ICML-03), 616–623.
4. Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830.
5. Hazarika, G., & Sharoff, S. (2019). Building NLP resources for low-resource languages of Northeast India: Challenges and strategies. Language Resources and Evaluation, 53(4), 745–768.
6. SEAlang Library. Tai Ahom Dictionary. Retrieved from https://sealang.net/ahom
7. Unicode Consortium. (2023). The Unicode Standard, Version 15.0, Tai Ahom Block U+11700–U+1173F.
8. Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media.