A Novel Approach to Sentiment Analysis using GMM-Enhanced N-gram LSTM Networks

K. Dhana Sree Devi
Department of CSE, GITAM School of Technology, GITAM Deemed to be University, Hyderabad Campus, Telangana, India
drdhanasree2@gmail.com (corresponding author)

V. Sireesha
Department of CSE, GITAM School of Technology, GITAM Deemed to be University, Hyderabad Campus, Telangana, India
svenduru@gitam.edu

C. Sudha
Department of CSE, GITAM School of Technology, GITAM Deemed to be University, Hyderabad Campus, Telangana, India
schinnap@gitam.edu

Malladi Ravisankar
Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur District, Andhra Pradesh, India
mravisankar@kluniversity.in

P. Dileep Kumar Reddy
Department of CSE, Narsimha Reddy Engineering College (Autonomous), Secunderabad, Telangana State, India
dileepreddy503@gmail.com

Received: 20 February 2025 | Revised: 11 March 2025 and 26 March 2025 | Accepted: 28 March 2025

 DOI: https://doi.org/10.48084/etasr.10640

ABSTRACT

Most e-commerce market platforms are improving their competitive benchmarks with continuously improving AI-based review analysis tools. Today, product review analysis is being prioritized from small to large companies to achieve parallel goals. Working with user text reviews that are coupled with diversifying sentiments, the market is now facing the real challenge of finding a perfect sentiment analysis approach that can meet business needs. This work presents a Gaussian Mixture Model (GMM) tokenizer to perform N-gram analysis on text. The proposed approach was compared with the LSTM baseline classifier on Amazon product reviews, and the experimental results showed that the GMM N-gram LSTM model outperformed the baseline LSTM. The accuracy of the proposed model was 85%, significantly better than the baseline LSTM (77%).

Keywords-tokenizer; sentiment analysis; text mining; NLP; n-gram; word vector; GMM; LSTM

I. INTRODUCTION

The expansion of e-commerce websites has led to the collection of huge amounts of user opinions through reviews on market products. Most business websites leverage these opinion-based reviews to build a sentiment analysis tool to track and analyze user buying patterns. Two different aspects are used to focus on user reviews: review context and grammar. In NLP, most documents are inferred with a weighted context rather than the usage of grammar, analyzing the text context and sentiment. The huge explosion of user-generated content has led to the development of many sentiment analysis approaches. Various studies on sentiment analysis have been presented, such as feature-driven, lexical-based, and rule-based approaches [1]. Feature-driven approaches use machine learning models to extract text features.

In [2], a promising rule-based approach was discussed to extract sentiments from text. In [3], a Rule-Based Model (RBM) was used to extract features for improved sentiment analysis. Extracting concept-level sentiments using dependency rules was discussed in [4]. In [5], various approaches to lexical-based sentiment analysis were discussed. In [6], a comparative study was performed on six lexicons. In [7], learn-based approaches were combined with lexicon analysis to enhance concept-level sentiment understanding. In [8], a lexicon-based scalable sentiment tool was discussed.

Text categorization using N-grams was discussed in [9], comparing the N-gram frequencies for text classification and showing better performance with varied text errors. In [10], an N-gram lexicon was presented to better understand text sentiments, combining unigrams with intensifiers to extract product review sentiments. In [11], the focus was on improved data extraction through N-grams, as not all N-grams provide abstract sentiment. In [12], a weighted approach was proposed to extract the most important N-grams. The adaptation of N-gram Gaussian Mixture Models (GMM) to sentiment analysis was discussed in [13, 14], showing that GMM clusters can transfer sentiments from one domain to another. In [15, 16], synthetic data generation used GMMs for sentiment analysis. In [17-19], computational efficiency was addressed from a variety of perspectives, introducing an online learning framework for EM-based estimation of GMMs.

In [20, 21], Twitter sentiments were examined with and without stop words. Many advanced NLP text mining approaches have emphasized the need for domain-specific stop words [22]. Some text mining approaches considered a stop-word filtering mechanism based on the most frequent words [23]. In [24], a comparative study was conducted on the effect of stop-word removal on extracting sentiments from movie reviews. In [25], various machine-learning approaches for sentiment analysis were discussed. In [26], sentiment extraction on Twitter data was presented using machine learning models. The empirical discussion in [27, 28] showed better performance of a Bayes model over a support vector model in sentiment classification. Deep learning techniques for sentiment analysis have been presented [29]. Sentiment analysis using convolutional neural networks was discussed in [30], while the impact of hyperparameters on LTSM was explored in [31].

Current research has evolved with many NLP approaches to understand sentiments through contexts using various text analytic methods. Among them, N-gram analysis has been more practiced for analyzing the text sentiments. The relevance of the word is better understood with N-gram analysis, as it aids in building a rich set of text features that could increase the performance of any ML or DL model. As the number of N-grams increases, the method suffers from data sparsity. Here, in the training data, a lot of N-grams could show up very infrequently or not at all. For unseen sequences, this leads to zero probability estimates, which yields predictions that are not accurate and weakens the model's ability to predict meaningful sequences. To address this problem, probabilistic models are coupled with N-gram analysis, and one such approach is using probabilistic GMMs.

GMMs present the probability distribution of N-grams as different distribution clusters. N-grams sharing the same contextual meaning fall into similar clusters, providing a better representation of data sparsity. This work took advantage of N-gram analysis using GMM models and trained the LSTM model for sentiment classification. GMM N-gram tokens were converted to sequence data to build the proposed approach. User reviews for a particular product were extracted from the Amazon website to build the dataset used in this study.

II. BACKGROUND STUDY AND THE PROPOSED APPROACH

Although there are many works on N-gram sentiment analysis, the proposed approach focuses on the N-gram analysis using a GMM tokenizer.

A. N-gram Tokenizer

N-gram modeling is an effective method for text analysis and NLP. N-gram analysis includes the analysis of N-word substrings of longer strings. N-gram analysis can also be character-based. To preserve the linguistic and contextual meaning of the text, word N-grams are mostly preferred. N-grams are most used in predictive analysis of text (to predict the next word).

Some single English words, such as happy, sad, and unhappy, will directly show off the hidden sentiment within them. But the real mesh is with words such as not, cannot, too, very, etc., which accompany single sentiment words. Their appearance with single direct sentiment words completely changes the sentiment of the text, resulting in huge misclassifications. A deeper analysis of such accompanied words can be achieved through word N-gram tokenization. N-gram tokenization is a method of separating words with a sequence of two or more.

B. Gaussian Mixture Model (GMM) Tokenizer

As a probabilistic function, the Gaussian mixture is a mixture of several Gaussians, where each Gaussian is a cluster and is identified by $c \in \{1, \dots, C\}$, where C is the number of clusters. Each Gaussian cluster has the following parameters:

- Mean μ, defining the Gaussian cluster center.
- Covariance parameter Σ, defining the width of the Gaussian cluster.
- A probability measure π that gives the size of the function.

GMMs are typical probabilistic models that work under the assumption that data is generated from a mixture of several Gaussian distributions, each with its own mean and covariance matrix. Figure 1 shows Gaussian distributions represented under three clusters, which can represent three text sentiment categories: negative, neutral, and positive. Each cluster is represented by a Gaussian distribution curve. Cluster 1, centered at point μ_1, shows the distribution of tokens with negative sentiments. Cluster 2, centered at point μ_2, shows the distribution of tokens with positive sentiments. This curve is slightly taller than the others, showcasing a smaller variance. Cluster 3, centered at point μ_3, shows the distribution of tokens

with neutral sentiments. The three Gaussian distributions partially overlap, particularly between adjacent clusters.

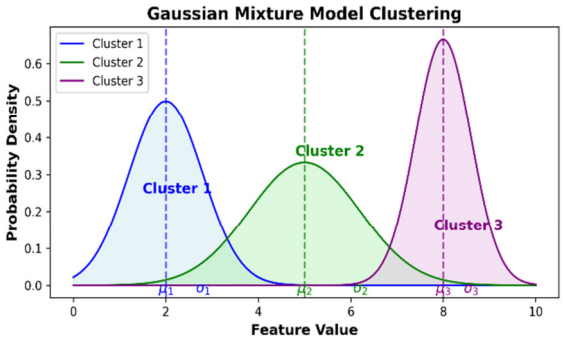

Fig. 1. GMM cluster distributions.

The GMM tokenization model works on a set of parallel GMMs, each acting as a tokenizer producing tokens of the highest probability. The Gaussian mixture statistical methods are applied to find the likelihood scores of each token. The GMM succeeds in mapping feature vectors to regions of acoustic space that correspond to relevant Gaussian components. The sentences are modeled by bigram models, in which the probability of sequences of two consecutive tokens is given by the following relation:

$$p(w_n|w_{n-1}) = \alpha_2 \cdot P(w_n|w_{n-1}) + \alpha_1 \cdot P(w_n) + \alpha_0 \quad (1)$$

where $\alpha_2 = 0.69$, $\alpha_1 = 0.35$ and $\alpha_0 = 0.01$ are fixed constants, and W_n and W_{n-1} are any two successive tokens.

C. Using the GMM for N-gram Tokenization

The usual N-gram tokenization is a rule-based approach that splits the text into fixed sequences by preserving the contextual meaning of the tokens. However, N-gram tokenization fails to understand the nuanced relationships between tokens, where contextual dependencies, polysemy, and collocations are frequently encountered. Previous studies have shown that the usual N-gram tokenization is less adaptable to varying text sentiments.

The GMM tokenizer overcomes all these disadvantages of the usual N-gram tokenizer using the probability distributions of the contextual words where the tokens can belong to multiple clusters. The token belonging to a relevant cluster has the highest probable score. This probabilistic approach captures the inherent ambiguity in NLP more effectively. GMM probabilities refer to the surrounding context, providing a more accurate tokenization by considering the probability distribution of neighboring tokens.

D. Proposed Approach for Sentiment Labeling

1) Data Cleaning and Feature Extraction from Text Reviews

The text reviews were cleaned and preprocessed by applying various feature engineering approaches such as stemming and lemmatization. After removing the stop-words, the corpus is subjected to bigram analysis to observe how the top sentiments vary. For example, for the sentence "the fox ran to the top of hill", the bigram tokenization is (the,fox),(ran,to),(the,top),(of,hill). From the cleaned text, reviews and numerical features, such as word embeddings, are extracted by giving the cleaned reviews input to the BERT model.

2) Modeling LSTM for Sentiment Labeling

LSTM was used for review sentiment classification. Many deep learning models have been introduced for this purpose [32]. LSTM outperformed by capturing long-distance dependencies of sequential data, just by introducing a cell stage that could preserve the state information for a long period of time. LSTM succeeded RNN by addressing the vanishing gradient issue. Variable-length sentences are transformed into vectors of fixed length by LSTN units. An LSTM unit takes input from six vectors at each time step h_{t-1} as shown below:

$$I_t = S_g(W_i * X_t + U_i * h_{t-1} + b_i) \quad (2)$$

$$F_t = S_g(W_f * X_t + U_f * h_{t-1} + b_f) \quad (3)$$

$$C_t^1 = tanh(W_c * X_t + U_c * h_{t-1} + b_c) \quad (4)$$

$$C_t = I_t * C_t^1 + F_t C_{t-1} \quad (5)$$

$$O_t = S_g(W_o * X_t + U_o * h_{t-1} + b_o) \quad (6)$$

$$h_t = O_t * relu(C_t) \quad (7)$$

where I_t, F_t, C_t and O_t are the input gates, forget gates, memory cells, and output gates of the LSTM at time t. W_i, W_f, W_c, W_o and U_i, U_f, U_c, and U_o are the weights corresponding to input X_t and hidden units h_{t-1}, in respective gates.

3) GMM N-Gram LSTM Word Embedding

The word embeddings that are used in the proposed LSTM are modeled by the GMM N-gram tokenizer. A review of length l is represented as a matrix (d, l) where each column is presented as a d-dimensional word embedding vector for each word. Here x is the feature vector. In general, LSTMs are modeled for time-based sequence data. As shown in Algorithm 1, the process takes an input sequence, a sequence of words, generates N-grams for $N = 2$, and constructs the embeddings.

```
Algorithm 1: Generate N-gram embeddings
Input a sequence of words:
W = [W_1, W_2, …, W_Ti] where T_i is the length of
the sequence.
For each token w_i, generate N-grams N =
[N_1, N_2, …, N_Tj], where T_j is the length of the
N-gram sequence.
Fit a Gaussian distribution on N to output
the GMM cluster distribution components.
Let C be the number of GMM components.
For each N-gram component, generate the
probability P(N_i | θ_j), where θ_j is the GMM
component.
Generate the word embeddings using the GMM
model given by
V_i = ∑_{j=1}^{C} P(N_i|θ_j)μ_j, where μ_j is the mean
vector of the j-th GMM component.
```

4) Procedure to Incorporate GMM Outputs into an LSTM

Given a review matrix l, the GMM N-gram tokenizer is applied to this input via a density function p. This model tokenizes the whole review with bigrams. The output of the GMM is a continuous probabilistic. A single Gaussian component has a density function given by:

$$p(x|\mu_k, \Sigma_k) = \frac{1}{((2\pi)^{\frac{d}{2}}|\sum k|1/2} e^{(-1/2(x-\mu_k)\top\Sigma_{k-1}(x-\mu_k)}) \quad (8)$$

where d is the dimensionality of x, and μ_k is the k-th Gaussian component.

As LSTM is modeled on sequence data, continuous data are discretized by passing the GMM output into a sliding window of size t to generate sequence data at regular time intervals, and the data is tabulated to be given input to LSTM. Figure 2 shows the three bigram GMM distributions. The contextual bigrams showing positive sentiment fall into the Positive GMM distribution, and negative bigrams fall into the Negative GMM distribution, as shown in Figure 2. Points belonging to multiple clusters show different probability scores based on relevancy.

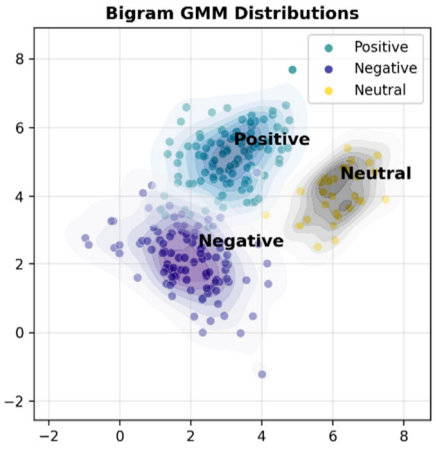

Fig. 2. Bigram GMM distributions showing three text sentiments.

5) Architecture and Training GMM N-gram LSTM Model

Figure 3 shows the architecture of the proposed model. Algorithm 2 shows the LSTM training using word embeddings. The cleaned text is passed into a BERT model for embedding vectors. These continuous vectors are then given to the GMM N-gram module to obtain the bigram sentiment distributions. The GMM N-gram model is trained to minimize the cross-entropy error:

$$L(x_i, y_i) = \sum_{j=1}^{k} 1\{y_i = j\}\ log(y_j^i) \quad (9)$$

which $1\{y_i = j\}$ is a conditional indicator, if true returns 1, otherwise 0.

On minimizing the cross-entropy loss, the output from the GMM N-gram module is converted to sequence data to be modeled by the LSTM classifier. A sliding window is operated with a size t to take the probabilistic continuous data and convert it to the required sequence data.

```
Algorithm 2: LSTM training with GMM
outputs.
1. Input the generated word embeddings
from the GMM V = {V_1, V_2, ..., V_T}.
2. Simulate LSTM to update the cell states
and the hidden states given by (2)-(7).
3. Output the sentiment from the final
hidden state h_t.
```

The LSTM model is trained with the BPTT algorithm to preserve the cell state. The SGD algorithm was used for parameter learning, along with ADAM for optimization.

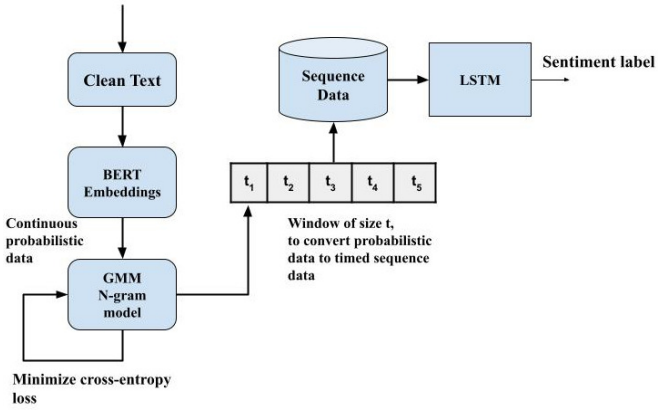

Fig. 3. GMM N-gram LSTM architecture.

The proposed GMM N-gram LSTM was trained using the parameters shown in Table I.

TABLE I. PROPOSED MODEL PARAMETERS

Model parameters	Value
Learning rate	0.01
epochs	50-100
Hidden layers	5
Optimizer	ADAM
Loss metric	Cross-entropy

III. EXPERIMENTAL RESULTS

The effectiveness of proposed the method was validated by comparing it with the traditional LSTM classifier.

A. Dataset

Experiments were carried out by initially scrapping reviews for an Amazon product, extracting nearly 7500 reviews. The dataset is available in [33]. The dataset included both long- and short-type reviews and was cleaned using various text preprocessing approaches. Finally, nearly 2500 short reviews were removed.

B. Performance Metrics

Two metrics were used to evaluate model performance, the ROC curve and the Precision-Recall curve, as the dataset is a little imbalanced.

- ROC curve: The ROC curve shows the trade-off between the true positive rate (sensitivity) and the false positive rate (1-specificity).
- Precision-Recall curve: The precision-recall curve shows the trade-off between precision (positive predictive value) and recall (sensitivity).

C. Experimental Results

The proposed GMM N-gram LSTM model was compared with a baseline LSTM that did not include GMM N-gram analysis. Figure 5 shows the increase of true positives when modeled using GMM N-grams. The accuracy of the proposed model was 85%, whereas that of the baseline LSTM was 77%, as shown in Figure 6.

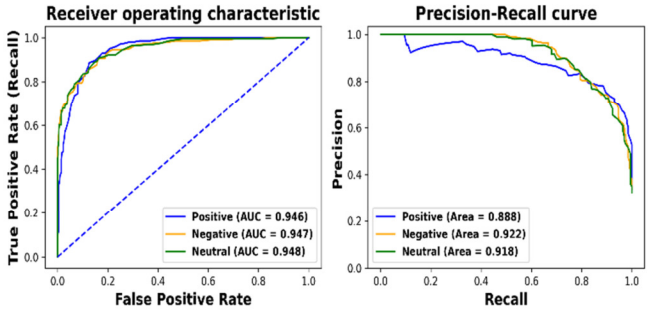

Fig. 4. Performance of baseline LSTM.

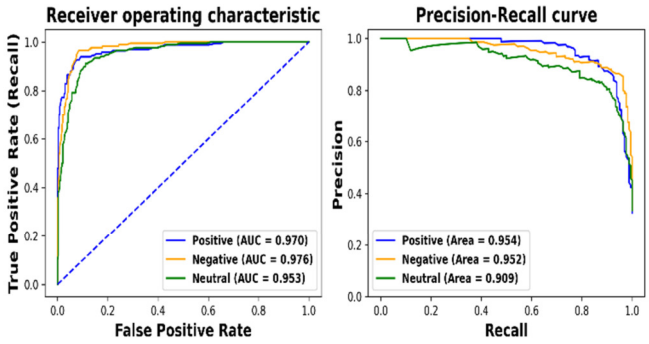

Fig. 5. Performance of GMM N-gram LSTM.

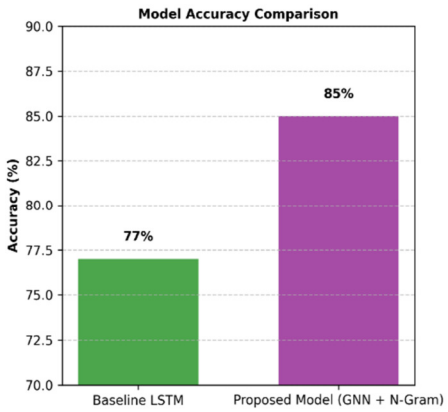

Fig. 6. GMM N-gram LSTM vs baseline LSTM.

IV. CONCLUSIONS

Today, businesses can take advantage of technology to establish stronger relationships with customers. To enhance this relationship, product review analysis is required. Many sentiment analysis tools have been proposed, but one way or another, these tools fail to give better recommendations. Working with varied user reviews with diverse sentiments, businesses now face the real challenge of a perfect sentiment analysis method that can meet their needs. This study used a GMM tokenizer to perform the N-gram analysis of text, where the resulting data were modeled by an LSTM classifier. The proposed approach was compared with a baseline LSTM classifier on Amazon product reviews, and the experimental results demonstrated that the GMM N-gram LSTM model outperformed the baseline LSTM model.

REFERENCES

[1] M. D. Devika, C. Sunitha, and A. Ganesh, "Sentiment Analysis: A Comparative Study on Different Approaches," *Procedia Computer Science*, vol. 87, pp. 44–49, Jan. 2016, https://doi.org/10.1016/j.procs.2016.05.124.

[2] J. Seror, "VADER Natural Language Processing in Market Sentiment Analysis," *SSRN Electronic Journal*, 2020, https://doi.org/10.2139/ssrn.3676706.

[3] M. T. F. A. Islami, A. R. Barakbah, and T. Harsono, "Social Media Engineering for Issues Feature Extraction using Categorization Knowledge Modelling and Rule-based Sentiment Analysis," *JOIV : International Journal on Informatics Visualization*, vol. 5, no. 1, pp. 83–93, Mar. 2021, https://doi.org/10.30630/joiv.5.1.397.

[4] S. Poria, E. Cambria, G. Winterstein, and G. B. Huang, "Sentic patterns: Dependency-based rules for concept-level sentiment analysis," *Knowledge-Based Systems*, vol. 69, pp. 45–63, Oct. 2014, https://doi.org/10.1016/j.knosys.2014.05.005.

[5] B. l. Velammal, "Development of knowledge based sentiment analysis system using lexicon approach on twitter data," *International Journal of Knowledge Management Studies*, vol. 10, no. 1, pp. 58–68, Jan. 2019, https://doi.org/10.1504/IJKMS.2019.097125.

[6] C. S. Khoo and S. B. Johnkhan, "Lexicon-based sentiment analysis: Comparative evaluation of six sentiment lexicons," *Journal of Information Science*, vol. 44, no. 4, pp. 491–511, Aug. 2018, https://doi.org/10.1177/0165551517703514.

[7] A. Mudinas, D. Zhang, and M. Levene, "Combining lexicon and learning based approaches for concept-level sentiment analysis," in *Proceedings of the First International Workshop on Issues of Sentiment Discovery and Opinion Mining*, New York, NY, USA, May 2012, pp. 1–8, https://doi.org/10.1145/2346676.2346681.

[8] C. Kaushik and A. Mishra, "A Scalable, Lexicon Based Technique for Sentiment Analysis." arXiv, Oct. 08, 2014, https://doi.org/10.48550/arXiv.1410.2265.

[9] J. Graovac, "A variant of n-gram based language-independent text categorization," *Intelligent Data Analysis*, vol. 18, no. 4, pp. 677–695, Jul. 2014, https://doi.org/10.3233/IDA-140663.

[10] A. Dey, M. Jenamani, and J. J. Thakkar, "*Senti-N-Gram*: An *n*-gram lexicon for sentiment analysis," *Expert Systems with Applications*, vol. 103, pp. 92–105, Aug. 2018, https://doi.org/10.1016/j.eswa.2018.03.004.

[11] S. Koshy and R. Padmajavalli, "Text Categorization of Multi-Label Documents For Text Mining," *International Journal of Data Mining Techniques and Applications*, vol. 4, no. 2, pp. 52–58, Dec. 2015, https://doi.org/10.20894/IJDMTA.102.004.002.001.

[12] M. Suzuki and S. Hirasawa, "Text Classification Using the Sum of Frequency Ratios of Word andN-gram Over Categories," *IEEJ Transactions on Electronics, Information and Systems*, vol. 129, pp. 118–124, Jan. 2009, https://doi.org/10.1541/ieejeiss.129.118.

[13] Q. Zhou, W. Zhou, and S. Wang, "Semantic adaptation network for unsupervised domain adaptation," *Neurocomputing*, vol. 454, pp. 313–323, Sep. 2021, https://doi.org/10.1016/j.neucom.2021.05.041.

[14] M. Wankhade, A. C. S. Rao, and C. Kulkarni, "A survey on sentiment analysis methods, applications, and challenges," *Artificial Intelligence Review*, vol. 55, no. 7, pp. 5731–5780, Oct. 2022, https://doi.org/10.1007/s10462-022-10144-1.

[15] P. Sundarreson and S. Kumarapathirage, "SentiGEN: Synthetic Data Generator for Sentiment Analysis," *Journal of Computing Theories and Applications*, vol. 1, no. 4, pp. 461–477, Apr. 2024, https://doi.org/10.62411/jcta.10480.

[16] A. R. Pathak, M. Pandey, and S. Rautaray, "Topic-level sentiment analysis of social media data using deep learning," *Applied Soft Computing*, vol. 108, Sep. 2021, Art. no. 107440, https://doi.org/10.1016/j.asoc.2021.107440.

[17] L. Xu and M. I. Jordan, "On Convergence Properties of the EM Algorithm for Gaussian Mixtures," *Neural Computation*, vol. 8, no. 1, pp. 129–151, Jan. 1996, https://doi.org/10.1162/neco.1996.8.1.129.

[18] O. Cappé and E. Moulines, "On-Line Expectation–Maximization Algorithm for latent Data Models," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 71, no. 3, pp. 593–613, Jun. 2009, https://doi.org/10.1111/j.1467-9868.2009.00698.x.

[19] J. J. Verbeek, N. Vlassis, and B. Kröse, "Efficient Greedy Learning of Gaussian Mixture Models," *Neural Computation*, vol. 15, no. 2, pp. 469–485, Feb. 2003, https://doi.org/10.1162/089976603762553004.

[20] B. Gunter, N. Koteyko, and D. Atanasova, "Sentiment Analysis: A Market-Relevant and Reliable Measure of Public Feeling?," *International Journal of Market Research*, vol. 56, no. 2, pp. 231–247, Mar. 2014, https://doi.org/10.2501/IJMR-2014-014.

[21] C. P. Li, L. H. Guo, and N. Lin, "Value Mining of Product Reviews Based on Sentiment Analysis," *Applied Mechanics and Materials*, vol. 713–715, pp. 2528–2531, 2015, https://doi.org/10.4028/www.scientific.net/AMM.713-715.2528.

[22] W. He, H. Wu, G. Yan, V. Akula, and J. Shen, "A novel social media competitive analytics framework with sentiment benchmarks," *Information & Management*, vol. 52, no. 7, pp. 801–812, Nov. 2015, https://doi.org/10.1016/j.im.2015.04.006.

[23] M. El Marrakchi, H. Bensaid, and M. Bellafkih, "Scoring reputation in online social networks," in *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA)*, Rabat, Oct. 2015, pp. 1–6, https://doi.org/10.1109/SITA.2015.7358420.

[24] R. Jayasanka, T. Madhushani, E. Marcus, I. Aberathne, and S. Premaratne, "Sentiment analysis for social media," in *Information Technology Research Symposium*, 2013, vol. 11, Art. no. 22.

[25] M. S. Akhtar, D. Gupta, A. Ekbal, and P. Bhattacharyya, "Feature selection and ensemble construction: A two-step method for aspect based sentiment analysis," *Knowledge-Based Systems*, vol. 125, pp. 116–135, Jun. 2017, https://doi.org/10.1016/j.knosys.2017.03.020.

[26] P. Patil and P. Yalagi, "Sentiment Analysis Levels and Techniques: A Survey," *International Journal of Innovations in Engineering and Technology*, vol. 6, no. 4, pp. 523–528, Apr. 2016.

[27] D. I. H. Farias and P. Rosso, "Irony, Sarcasm, and Sentiment Analysis," in *Sentiment Analysis in Social Networks*, F. A. Pozzi, E. Fersini, E. Messina, and B. Liu, Eds. Boston, MA. USA: Morgan Kaufmann, 2017, pp. 113–128.

[28] J. Zhu, H. Wang, B. K. Tsou, and M. Zhu, "Multi-aspect opinion polling from textual reviews," in *Proceedings of the 18th ACM conference on Information and knowledge management*, Aug. 2009, pp. 1799–1802, https://doi.org/10.1145/1645953.1646233.

[29] D. Vilares, C. Gómez-Rodríguez, and M. A. Alonso, "Universal, unsupervised (rule-based), uncovered sentiment analysis," *Knowledge-Based Systems*, vol. 118, pp. 45–55, Feb. 2017, https://doi.org/10.1016/j.knosys.2016.11.014.

[30] S. Kumar Singh, P. Verma, and P. Kumar, "Sentiment Analysis Using Machine Learning Techniques on Twitter: A Critical Review," *Advances in Mathematics: Scientific Journal*, vol. 9, no. 9, pp. 7085–7092, Aug. 2020, https://doi.org/10.37418/amsj.9.9.58.

[31] I. A. Kandhro, S. Z. Jumani, F. Ali, Z. U. Shaikh, M. A. Arain, and A. A. Shaikh, "Performance Analysis of Hyperparameters on a Sentiment Analysis Model," *Engineering, Technology & Applied Science Research*, vol. 10, no. 4, pp. 6016–6020, Aug. 2020, https://doi.org/10.48084/etasr.3549.

[32] M. R. R. Rana, A. Nawaz, T. Ali, A. M. El-Sherbeeny, and W. Ali, "A BiLSTM-CF and BiGRU-based Deep Sentiment Analysis Model to Explore Customer Reviews for Effective Recommendations," *Engineering, Technology & Applied Science Research*, vol. 13, no. 5, pp. 11739–11746, Oct. 2023, https://doi.org/10.48084/etasr.6278.

[33] K. D. S. Devi, "dhanasreek/Web_Scrapping." Jan. 03, 2022, [Online]. Available: https://github.com/dhanasreek/Web_Scrapping.