A Comparative Study of Naïve Bayes, SVM, Random Forest, and LSTM Performance in Sentiment Analysis on a Movie Review Dataset

Widi Widayat
Faculty of Communication and Informatics, Informatics Department
Universitas Muhammadiyah Surakarta, Indonesia
widi.widayat@ums.ac.id

Submitted: August, 02, 2025, accepted: May, 05, 2026

Abstract

Sentiment analysis is an important task in natural language processing that identifies and classifies opinions or emotions in textual data. This study compares the performance of four classification algorithms—Naïve Bayes, Support Vector Machine (SVM), Random Forest, and Long Short-Term Memory (LSTM)—on 25,000 English-language movie reviews with balanced sentiment labels. Text preprocessing includes cleaning, tokenization, and TF-IDF vectorization for traditional models. For LSTM, both randomly initialized embeddings and pre-trained embeddings are tested. Results, evaluated using accuracy, F1-score, and confusion matrix, show that SVM performs best with 89% accuracy, followed by Naïve Bayes and LSTM at 86%, and Random Forest at 82%. LSTM performs poorly with TF-IDF or self-trained embeddings but improves significantly with pre-trained embeddings. These findings indicate that traditional models, especially SVM, remain highly effective for sentiment analysis on moderately sized datasets, while LSTM requires proper text representation to perform competitively. With its strong performance, SVM can be practically applied in real-world scenarios such as customer feedback monitoring, product review classification, and public opinion analysis in digital platforms.

Keywords*: sentiment analysis, text classification, naïve bayes, svm, long short-term memory (lstm)*

1. Introduction

Sentiment analysis has emerged as a critical task in natural language processing (NLP), enabling systems to detect and classify subjective opinions or emotional content within textual data. This technique plays a pivotal role in various applications, including product review mining, social media monitoring, and customer experience analysis [1][2]. The growing volume of user-generated content on digital platforms has heightened the demand for reliable sentiment classification methods that can handle large and diverse datasets effectively. A fundamental challenge in sentiment analysis is determining the most suitable classification method that balances accuracy and the ability to generalize across diverse datasets. A variety of machine learning techniques have been explored in previous studies, including Naïve Bayes [3], Support Vector Machine (SVM) [4], and Random Forest [5], which have been widely adopted due to their simplicity and interpretability. In parallel, deep learning approaches such as Long Short-Term Memory (LSTM) have gained increasing attention for their ability to capture long-range dependencies in textual data [6][7][8][9].

Recent comparative studies have benchmarked these algorithms across various domains. Basri et al. (2024) evaluated Naïve Bayes, SVM, Decision Trees, and Random Forest on usability testing review data, finding Random Forest to yield the highest accuracy. Comparative analysis on product and restaurant reviews similarly showed that traditional models often outperform or match deep learning baselines when data volumes are moderate [9], [10]. Kayed et al. (2023) provided a comprehensive survey of over 100 deep learning models across domains including movie reviews, demonstrating that model performance is heavily influenced by dataset size and preprocessing factors [7].

As public opinion data becomes increasingly accessible, research in the field of sentiment analysis has experienced rapid growth. Many trending and widely discussed topics in society have become relevant and attractive subjects for sentiment analysis studies. One of the most commonly used sources for gathering opinion data is the social media platform Twitter (now known as X), which provides

a continuous stream of user-generated content. Several studies have utilized Twitter data for sentiment classification, including a study by Damayanti and Kemas (2024), which analyzed public opinion related to the 2024 presidential election. The study by Damayanti and Kemas (2024) demonstrated that sentiment analysis of the 2024 presidential election using the SVM algorithm with Word2Vec feature extraction achieved an accuracy of 90.75%, outperforming previous studies employing the same method [11]. Another study by Lenggo Geni et al. utilized the IndoBERT model to analyze public sentiment on Twitter regarding the 2024 Indonesian general election, revealing that the majority of tweets expressed neutral sentiment (83.7%) and that IndoBERT outperformed machine learning and lexicon-based approaches, achieving an accuracy of 83.5% [12]. Other topics, such as government political policies that have sparked widespread public discussion, also present interesting areas to be explored through sentiment analysis. These include issues related to the new capital city (IKN) [13], the pre-employment card program [14], and the Omnibus Law on Job Creation [15]. Sentiment analysis provides a valuable means for evaluating and analyzing public opinion on specific policy-related topics.

Meanwhile, in the domain of movie review sentiment analysis, several studies have utilized datasets consisting of 25,000 movie reviews to evaluate deep learning models such as LSTM and transformer-based approaches. For instance, models like LSTM with Word2Vec embeddings have shown promising results, while fine-tuned BERT combined with BiLSTM has achieved accuracy as high as 97.67% on the IMDb dataset—albeit with the need for large-scale pretraining [16].

Despite the growing number of individual model evaluations, there remains a lack of direct comparative studies that systematically evaluate Naïve Bayes, SVM, Random Forest, and LSTM using the same dataset, consistent preprocessing techniques, and uniform evaluation metrics. Existing studies often focus on a subset of models, employ different datasets, or apply varied preprocessing pipelines, making it difficult to draw fair and reliable comparisons. This methodological inconsistency highlights a research gap that needs to be addressed to better understand the practical trade-offs between traditional machine learning and deep learning approaches. Therefore, this study aims to conduct a comprehensive evaluation of all four models using a standardized dataset of 25,000 English-language movie reviews. Each model is trained and tested using identical preprocessing steps—including text cleaning, tokenization, and vectorization—and is assessed based on accuracy, F1-score, and confusion matrix. The results are expected to provide clearer insights into the relative strengths and limitations of each approach within a unified experimental framework.

2. Method

This research is designed to conduct a comparative performance evaluation of four classification algorithms—Naïve Bayes, Support Vector Machine (SVM), Random Forest, and Long Short-Term Memory (LSTM)—within the context of sentiment analysis on movie review data. The methodology is structured into five sequential phases: (1) dataset acquisition and preparation, (2) preprocessing of text data, (3) feature vectorization, (4) model training, and (5) performance evaluation. An overview of this workflow is illustrated in Fig. 1, which depicts the sequential steps carried out in this study.

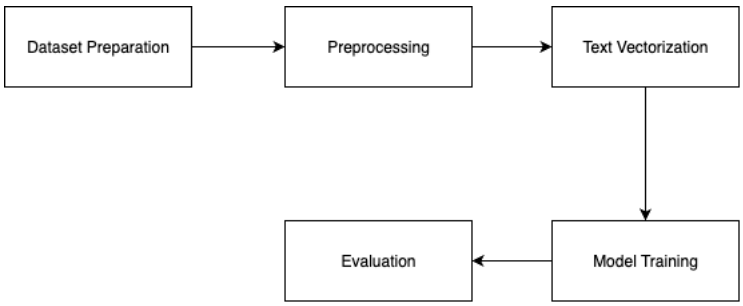

Fig 1. Research Workflow Diagram

Fig. 1 provides a structured overview of the research methodology, encapsulating the five core stages implemented in this study: data collection, preprocessing, feature vectorization, model training, and evaluation. This visual aid supports a clearer understanding of the experimental workflow and the comparative analysis conducted across the four classification algorithms.

2.1. Dataset Preparation

The dataset employed in this study comprises 25,000 English-language movie reviews, equally distributed between two sentiment classes: positive (12,500 samples) and negative (12,500 samples). Each entry in the dataset consists of a text review and a binary sentiment label, where 1 represents a positive sentiment and 0 denotes a negative one. This balanced composition ensures the absence of class imbalance bias and enhances the reliability of performance comparison across different models. The dataset is a standardized benchmark, widely adopted in prior sentiment classification research, particularly for IMDb movie review tasks. A summary of the Movie review dataset's general characteristics, including word counts and vocabulary size, is presented in Table 1 below.

Table 1. Movie Review Dataset

Dataset Name	Total Number of Files/Reviews	Positive Review	Negative Review	Total Number of Words	Unique Words	Average Words per Review
Movie Review	25.000	12.500	12.500	5.844.680	73.736	233,79

Table 1 summarizes the dataset's composition and linguistic properties, highlighting its balanced class distribution and lexical complexity.

2.2. Preprocessing

The dataset employed in this study comprises 25,000 English-language movie reviews, equally distributed between two sentiment classes: positive (12,500 samples) and negative (12,500 samples). Each entry in the dataset consists of a text review and a binary sentiment label, where 1 represents a positive sentiment and 0

Before any modeling can be performed, the raw text data must undergo preprocessing to normalize the input and remove noise. The preprocessing pipeline includes several standardized operations:

a. Lowercasing: All text is converted to lowercase to eliminate discrepancies between word tokens such as "Good" and "good."
b. Punctuation and Special Character Removal: All non-alphanumeric characters are stripped to retain only meaningful text.
c. Stopword Elimination: Common but non-informative words (e.g., "the," "is," "and") are removed using a predefined stopword list.
d. Tokenization: Each review is broken down into individual word tokens to facilitate further analysis.

For the LSTM model, additional preprocessing steps were applied:

a. Sequence Padding: All tokenized reviews were padded or truncated to a uniform sequence length to ensure compatibility with fixed-length LSTM input requirements.

These preprocessing steps were applied uniformly across the dataset to ensure that each classification model receives input under the same data conditions.

2.3. Text Vectorization

After preprocessing, the textual data must be transformed into numerical representations suitable for machine learning algorithms. The vectorization strategy was selected based on the type of model used:

a. For Naïve Bayes, SVM, and Random Forest: The Term Frequency-Inverse Document Frequency (TF-IDF) technique was used. This method weighs each term's importance in a document relative to its frequency across the corpus, producing a sparse matrix of numerical features representing each review.
b. For the LSTM model: Instead of TF-IDF, word embeddings were used via Keras's Embedding layer. This approach maps each word token to a dense vector of fixed dimensions, capturing semantic similarity between words. A vocabulary size threshold was applied to exclude rare words, and padding was again used to maintain uniform input length.

These vectorization strategies ensure compatibility with each model while preserving relevant semantic information in the reviews.

2.4. Model Training

Each classification model was trained independently using a consistent data split strategy, where 80% of the dataset was allocated for training and 20% for testing. To ensure fair comparison, no data augmentation or external datasets were used. The following configurations were applied:

a. Naïve Bayes: Implemented using the Multinomial Naïve Bayes variant, which is well-suited for discrete features like word counts or TF-IDF.
b. SVM: The model employed a linear kernel, with the regularization parameter C set to 1.0, which is commonly used as a baseline. Hyperparameters were validated using standard techniques to achieve optimal performance on the TF-IDF feature space.
c. Random Forest: A forest of 100 decision trees was constructed, with default depth settings and bootstrapping enabled.
d. LSTM: The architecture included an embedding layer, a single LSTM layer with 128 units, a dropout layer 0.2 to prevent overfitting, and a dense output layer with a sigmoid activation function for binary classification.

All models were trained using identical training data to maintain experimental consistency.

2.5. Evaluation

Model performance was assessed using three standard classification metrics. Accuracy, the ratio of correctly predicted samples to the total number of test samples. F1-Score, the harmonic mean of precision and recall, particularly useful for evaluating imbalanced classes or false-positive sensitivity. Confusion Matrix, a tabular representation showing true positives, false positives, true negatives, and false negatives—providing insight into types of classification errors.

These metrics provide a comprehensive view of each model's effectiveness in identifying sentiment polarity. All evaluations were conducted on the same test subset, ensuring a controlled environment for unbiased model comparison.

3. Result and Discussion

This section presents the experimental results of the four classification models—Naïve Bayes, Support Vector Machine (SVM), Random Forest, and Long Short-Term Memory (LSTM)—on the sentiment analysis task using the IMDb movie review dataset. The evaluation was conducted using three primary performance metrics: accuracy, F1-score, and confusion matrix.

3.1. Dataset

The dataset used in this study was a publicly available collection of 25,000 English-language movie reviews, balanced equally between positive and negative sentiments. This dataset was selected due to its widespread use in sentiment analysis research, making it an appropriate benchmark for evaluating and comparing the performance of various machine learning models. Each record in the dataset was stored in a relational table named *Movie Review*, which comprised two primary attributes: the texts column, containing the full text of each review, and the label column, an integer field indicating sentiment polarity, where a value of 1 represented a positive sentiment and 0 represented a negative sentiment.

The dataset was already labeled and curated, eliminating the need for manual annotation. Furthermore, an initial inspection confirmed that there were no missing or duplicate entries, indicating that the dataset was clean and ready for use without extensive preprocessing. One of the key reasons for selecting this dataset was its balanced class distribution, with an equal number of positive and negative reviews, which helped mitigate bias during model training. In addition, the dataset's size, consisting of 25,000 records, was sufficiently large for training both traditional machine learning algorithms and deep learning models, while still maintaining manageable computational requirements. Movie reviews, as the primary content, were particularly well-suited for sentiment classification tasks due to their rich use of subjective language and expressions of opinion.

Originally, the dataset was provided in the form of individual plain text files (txt files), where each file contained a single review. To facilitate more efficient processing and analysis, these text files were programmatically converted into a structured JSON format. This conversion allowed the data to be more easily parsed and organized into a DataFrame, in which each entry included two key fields: one for the textual content of the review and another for the sentiment label. The resulting DataFrame format enabled streamlined preprocessing and was used as the standardized input for all subsequent experiments.

Overall, this structured and well-balanced dataset provided a robust foundation for evaluating the performance of the sentiment classification models employed in this study.

Shown below is a sample review file categorized under the positive sentiment class.

```
{'sentiment': 'positive',
  'texts': "So you think a talking parrot is not your cup of tea huh? Well,
think again. Paulie is a wonderful film filled with touching moments.The
characters are all lovable especially Paulie as he enters the lives of many
people on his journey.It is journey worth experiencing. Don't miss it! It is
available on home video."},
```

In contrast to tweet data, which is constrained by a maximum length of 280 characters, the movie review dataset comprises reviews of varying lengths, with an average of approximately 234 words per review.

3.2. Data Preprocessing

Before being used as input to the machine learning models, the review data underwent a series of preprocessing steps to improve data quality and ensure consistency across all samples. The first step was text cleaning, which involved the removal of unwanted elements such as HTML tags, URLs, and other non-alphanumeric symbols that did not contribute meaningful information to the sentiment analysis task. This was followed by lowercasing, where all text was converted into lowercase to avoid treating the same words in different cases (e.g., “Good” and “good”) as distinct tokens.

Next, all punctuation marks and special characters were removed to reduce noise and simplify the textual data. To further focus the models on informative content, stopwords—which are common words such as “the”, “is”, and “and” that carry minimal semantic value—were eliminated using a standard stopword list. After that, the reviews were tokenized, splitting each sentence into a sequence of individual words, which is a crucial step for transforming text into numerical representations.

For models that require fixed-length input such as LSTM, the resulting token sequences were then padded or truncated to a consistent length. This ensured that all input data had the same shape regardless of the original review length. Finally, the sentiment labels were encoded into binary integers, where 1 represented a positive sentiment and 0 represented a negative sentiment. These preprocessing steps were implemented using well-established natural language processing libraries such as NLTK and Keras, transforming the raw text into a clean and structured format ready for both classical and deep learning models.

3.3. Text Representation

To enable the machine learning models to effectively interpret the textual data, two different text representation techniques were applied, tailored to the model types used in this study. For the classical machine learning models—Naïve Bayes, Support Vector Machine (SVM), and Random Forest—the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization technique was employed. This method converted each review into a weighted numerical vector, where the weight of each term reflected its importance in the specific document relative to its frequency across the entire corpus. By doing so, the models were able to emphasize more distinctive and informative terms, thereby enhancing classification accuracy.

In contrast, the LSTM model utilized a word embedding approach to represent text in a dense and semantically meaningful form. Each word was first tokenized into integer sequences using Keras’ tokenizer, and then mapped into vector representations through an embedding layer. Specifically, this study employed pretrained word embeddings from GloVe (Global Vectors for Word Representation), using the glove.6B.100d version. This version consists of 100-dimensional word vectors trained on a 6-billion-token corpus from Wikipedia and Gigaword. GloVe is an unsupervised learning algorithm developed by Stanford University that generates word embeddings by aggregating global word-word co-occurrence statistics. Unlike context-limited models, GloVe captures both syntactic and semantic relationships between words by learning a continuous vector space in which semantically similar words are located near each other. Leveraging these pretrained embeddings allowed the LSTM model to benefit from rich linguistic knowledge, which contributed to improved contextual understanding and more robust sentiment classification.

3.4. Naïve Bayes

The Naïve Bayes classifier was implemented as one of the baseline models for sentiment classification in this study. This algorithm is widely used in text classification tasks due to its simplicity, efficiency, and strong performance on high-dimensional data such as textual features. It operates under the assumption of conditional independence between features, which, despite being a simplification, often leads to robust results in practice.

For this study, the Multinomial Naïve Bayes variant was selected, as it is particularly well-suited for discrete data like term frequencies or TF-IDF scores. Prior to modeling, the textual data were vectorized using TF-IDF (Term Frequency–Inverse Document Frequency), which transformed each document into a sparse numerical vector representing the importance of each word relative to the entire corpus.

The model was trained on the vectorized training set, and hyperparameters such as the smoothing parameter (α) were tuned using cross-validation to avoid overfitting and improve generalization. After training, the model was evaluated on the test set using standard classification metrics, including accuracy, precision, recall, and F1-score.

In order to improve model performance and determine the most effective Naïve Bayes configuration, a series of experiments were conducted by varying the value of the max_features parameter used in the TF-IDF vectorization process. Specifically, the values tested were 500, 1,000, 3,000, 5,000, 7,000, 10,000, and 15,000.

The experimental results demonstrated a positive correlation between the number of features and classification accuracy. As the number of features increased, the model's ability to capture relevant patterns in the data also improved. The best performance was observed when using 15,000 features, achieving an accuracy of 86%. Fig. 2 presents a visual comparison of the Naïve Bayes model's performance across different max_features settings. Fig. 2 demonstrates that increasing the max_features value results in higher model accuracy. This finding suggests that a larger number of features helps the model extract more meaningful information, thereby enhancing its overall performance.

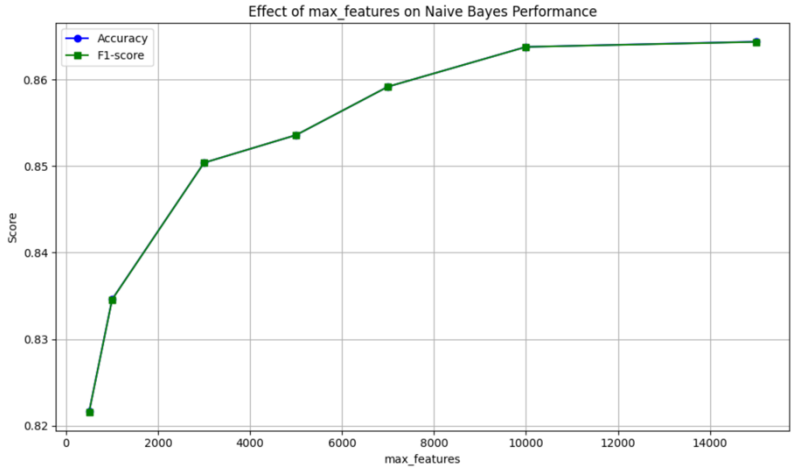

Fig 2. Naïve Bayes performance in relation to different max feature sizes

3.5. SVM

In this study, the Support Vector Machine (SVM) classifier was also employed to evaluate its performance in sentiment classification. Similar to the Naïve Bayes experiments, a series of trials were conducted by varying the max_features parameter in the TF-IDF vectorization process. The values used were 500, 1000, 3000, 5000, 7000, 10,000, and 15,000. This variation aimed to identify the optimal number of features that could yield the best performance for the SVM model.

The results demonstrated that SVM consistently outperformed Naïve Bayes across different feature sizes. The classification performance improved as the number of features increased, with the highest accuracy reaching 89% when max_features was set to 15,000. This suggests that SVM is more effective in handling high-dimensional sparse data representations such as TF-IDF vectors.

The Fig. 3 below illustrates the performance of the SVM model compared to different values of the max_features parameter. The increase in max_features provided the SVM with a broader vocabulary, enhancing its ability to distinguish subtle sentiment patterns. This improvement highlights SVM's strength in managing high-dimensional feature spaces derived from TF-IDF representations.

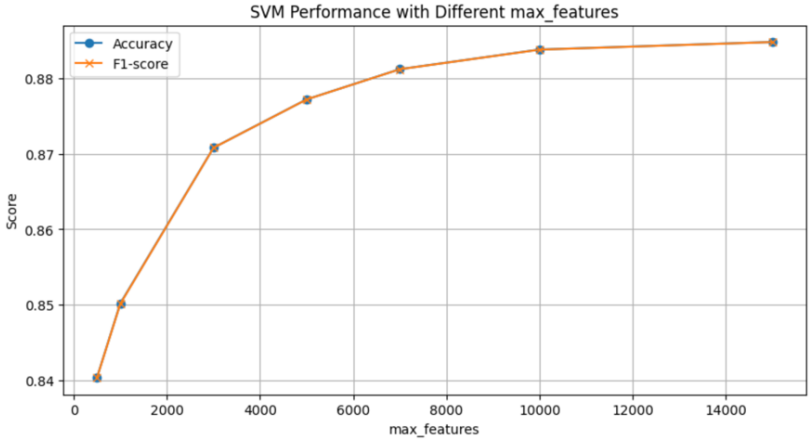

Fig 3. SVM performance in different max feature sizes

The Fig. 4 below compares the performance of the Naïve Bayes and SVM models.

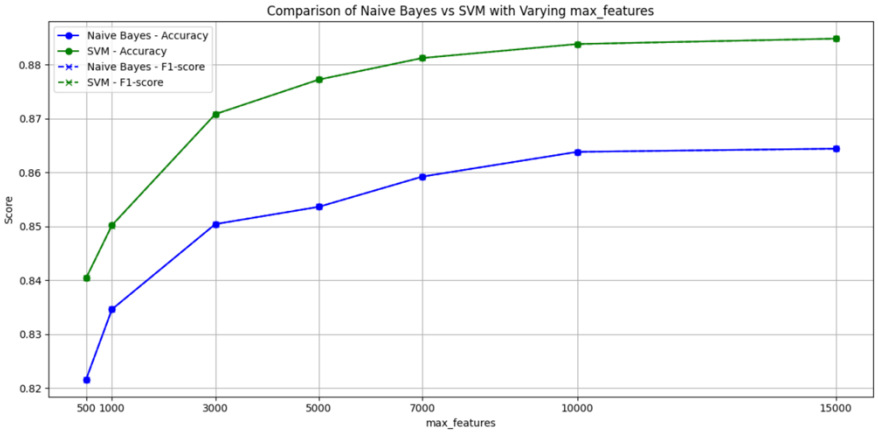

Fig 4. Performance comparison between Naïve Bayes and SVM

As illustrated in the Fig. 4 above, the performance of the SVM model consistently outperforms that of the Naïve Bayes model, demonstrating a clear and significant advantage.

3.6. Random Forest

In this study, the Random Forest classifier was also employed to evaluate its performance in sentiment classification. Similar to the Naïve Bayes and SVM models, the input text data were transformed using TF-IDF vectorization. To determine the optimal configuration, a series of experiments were conducted by varying the max_features parameter, using the same range as applied in previous models: 500, 1000, 3000, 5000, 7000, 10000, and 15000.

The results indicated that the performance of the Random Forest model improved as the number of features increased, reaching its highest accuracy at a max_features value of 15000, with a peak performance of 85%. This suggests that Random Forest benefits from a richer feature set when applied to text classification tasks, allowing it to capture more relevant patterns from the data.

The following Fig. 5 illustrates the performance trend of the Random Forest model across different max_features values.

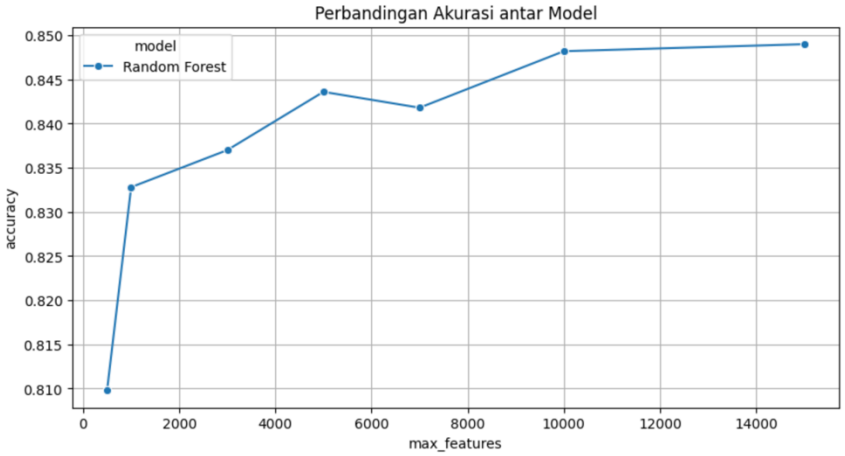

Fig 5. Random Forest performance in different max feature sizes

The following Fig. 6 presents a comparative performance analysis of the three classical machine learning models: Naïve Bayes, Support Vector Machine (SVM), and Random Forest. As illustrated in the figure, SVM outperforms both Naïve Bayes and Random Forest in terms of classification accuracy, demonstrating its superior capability in handling the sentiment analysis task on this dataset.

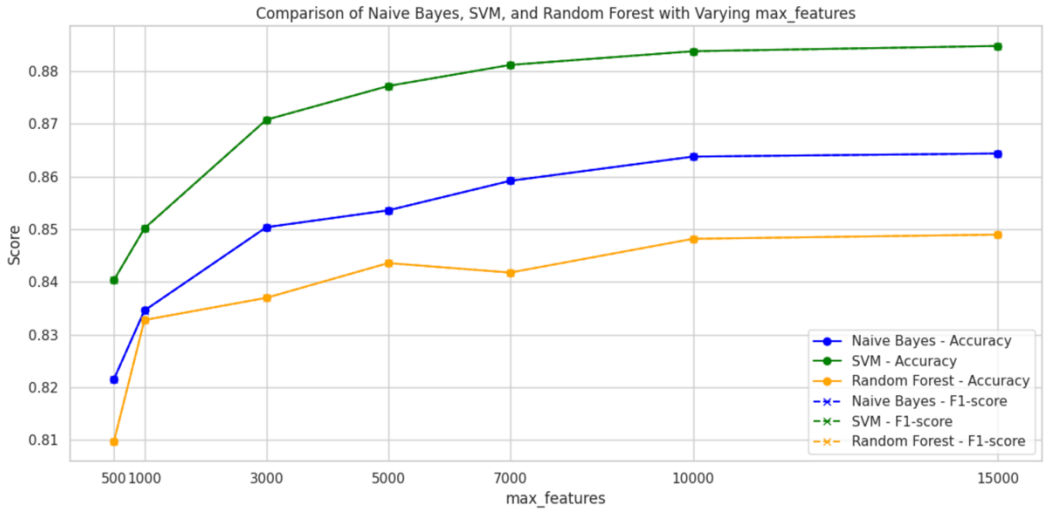

Fig 6. Comparative Performance of Naïve Bayes, SVM, and Random Forest Models

3.7. LSTM

In addition to classical machine learning models, a Long Short-Term Memory (LSTM) neural network was employed to capture the sequential and contextual information present in textual data. Prior to training, each review was tokenized and converted into a sequence of integers using Keras' Tokenizer. These sequences were then padded to ensure uniform input length across all samples.

Several preliminary experiments were conducted to determine the most suitable input representation for the LSTM model. Initially, text vectorization using the TF-IDF approach was tested, followed by the use of an embedding layer initialized randomly and trained solely on the dataset. However, these configurations yielded suboptimal results, indicating that the model struggled to effectively capture the semantic relationships between words with limited training data.

To address this, an alternative strategy was adopted by incorporating a pre-trained word embedding. Specifically, the GloVe.6B.100d embedding was used to initialize the embedding layer. GloVe (Global Vectors for Word Representation) generates 100-dimensional word vectors trained on a large corpus (Wikipedia and Gigaword), capturing global word-word co-occurrence statistics. By using this pre-trained embedding (set as non-trainable), the LSTM model could leverage rich semantic information learned from a broader context.

The final architecture consisted of an Embedding layer initialized with GloVe weights, an LSTM layer with 128 units, a Dropout layer to prevent overfitting, and a Dense output layer with sigmoid activation for binary classification. The model was compiled using the binary cross-entropy loss function and the Adam optimizer.

After training, the LSTM model achieved an accuracy of 86%, which, while competitive, was slightly lower than the best-performing classical model (SVM). This finding suggests that although LSTM has the capacity to capture complex sequential dependencies, its effectiveness can be heavily influenced by the choice of text representation and the availability of extensive training data.

The Fig. 7 below illustrates the performance comparison of the four models using the TF-IDF technique for text vectorization. It is evident that the LSTM model performs poorly in this setting. This result can be attributed to the fact that LSTM models are more effective when they can leverage the sequential and contextual nature of textual data directly, which is not well captured by TF-IDF representations.

Fig. 8 demonstrates the progressive improvement in the LSTM model's performance over successive training epochs. In this experiment, the model was trained for 10 epochs, with the highest accuracy achieved at 86%.

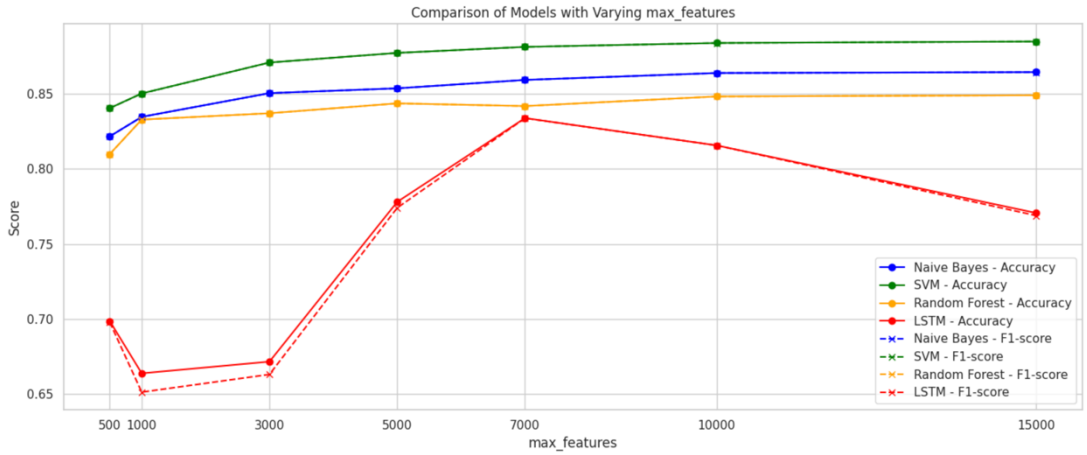

Fig 7. Comparative Performance of Naïve Bayes, SVM, Random Forest, and LSTM Models

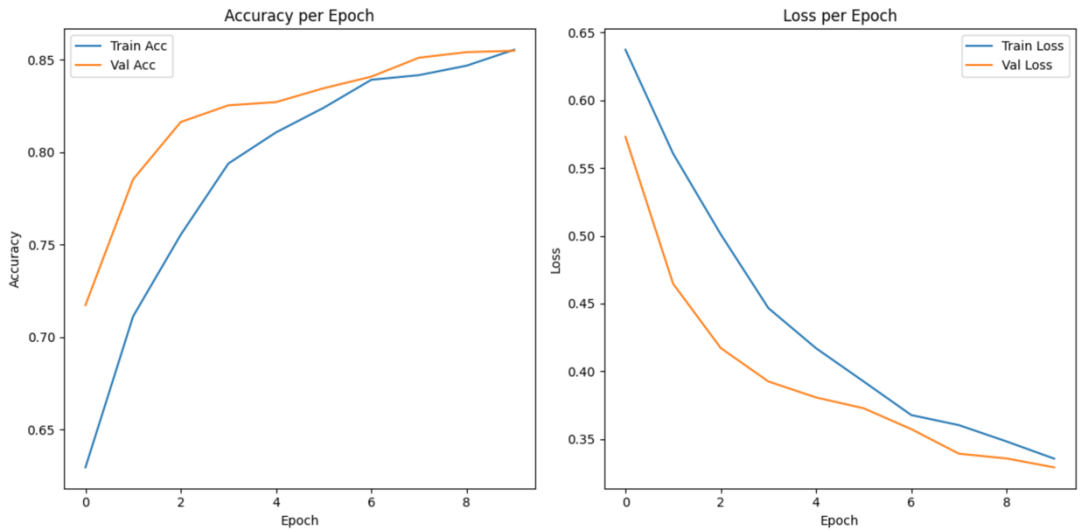

Fig 8. LSTM performance each epoch

To comprehensively evaluate the effectiveness of each model, a comparative summary is presented in Table 2, highlighting key performance metrics such as accuracy and F1-score.

Table 2. Performance Comparison of Classification Models on Movie Review Dataset

Model	Vectorization	Pre-trained	Accuracy
Naïve Bayes	TF-IDF	No	86%
SVM	TF-IDF	No	89%
Random Forest	TF-IDF	No	85%
LSTM	Word Embedding (Glove)	Yes	86%

Table 2 summarizes the performance of each classification model evaluated in this study. All traditional machine learning models—Naïve Bayes, SVM, and Random Forest—used TF-IDF vectorization and were trained without any pre-trained embeddings. Among these, the SVM model achieved the highest accuracy at 89%, indicating its robustness in handling linearly separable features generated by TF-IDF. Meanwhile, Naïve Bayes and Random Forest reached 86% and 85% accuracy, respectively.

Interestingly, the LSTM model, which utilized pre-trained word embeddings (GloVe), achieved an accuracy of 86%, comparable to Naïve Bayes. Despite the advantage of leveraging contextual semantic information through word embeddings, LSTM did not significantly outperform the simpler models. This suggests that while deep learning approaches offer more expressive representations, their effectiveness is still highly dependent on factors such as hyperparameter tuning, dataset size, and computational resources.

4. Conclusion

This study aimed to evaluate and compare the performance of four machine learning models—Naïve Bayes, Support Vector Machine (SVM), Random Forest, and Long Short-Term Memory (LSTM)—in sentiment analysis of movie reviews. A dataset consisting of 25,000 pre-labeled English-language reviews was utilized, with a balanced distribution of positive and negative sentiments.

Preprocessing steps such as text cleaning, lowercasing, punctuation and stopword removal, as well as lemmatization, were applied to standardize and prepare the textual data. Text vectorization was carried out using TF-IDF for the traditional models (Naïve Bayes, SVM, Random Forest), and pre-trained word embeddings (GloVe 6B with 100-dimensional vectors) for the LSTM model to capture semantic meaning.

The results revealed that the SVM model consistently outperformed the other models, achieving the highest accuracy of 89%, followed closely by the LSTM and Naïve Bayes models, each with an accuracy of 86%. Random Forest showed relatively lower performance, with an accuracy of 85%. Furthermore, although LSTM initially yielded suboptimal results using traditional vectorization or randomly initialized embeddings, significant performance improvements were observed when employing pre-trained word embeddings.

In conclusion, SVM proved to be the most effective model for sentiment classification in this study, particularly when paired with TF-IDF vectorization. Meanwhile, the LSTM model demonstrated competitive performance when supported by semantic-rich embeddings. These findings highlight the importance of appropriate feature representation in optimizing model performance and suggest that hybrid approaches integrating both linguistic context and statistical weighting may yield even better outcomes in future research.

Given its simplicity, scalability, and robust performance, the SVM model can be effectively applied for large-scale sentiment analysis tasks in various domains such as customer feedback systems or social media monitoring. Moreover, future research can explore enhancements to the LSTM model through fine-tuning on domain-specific corpora or combining it with attention mechanisms to further boost its performance.

Reference

[1] B. Liu, *Sentiment Analysis and Opinion Mining*. San Rafael, CA, USA: Morgan & Claypool, 2012.

[2] B. Pang and L. Lee, “Opinion mining and sentiment analysis,” *Found. Trends Inf. Retr.*, vol. 2, no. 1–2, pp. 1–135, 2008.

[3] H. Basri, M. B. S. Junianto, and I. Kusyadi, “Enhancing Usability Testing Through Sentiment Analysis: A Comparative Study Using SVM, Naive Bayes, Decision Trees and Random Forest,” *J. Teknol. Sistem Inform. Apl.*, vol. 7, no. 4, pp. 1603–1610, Oct. 2024.

[4] N. Febriyanti and A. F. Rozi, “Komparasi Algoritma Naïve Bayes, Support Vector Machine, dan Random Forest untuk Analisis Sentimen Ulasan Pengguna Aplikasi CGV Cinemas Indonesia,” *BITS*, vol. 7, no. 1, pp. 27–34, 2022.

[5] L. Breiman, “Random Forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[6] K. Pandit, A. Patel, R. Shah, and S. Jain, “Comparative Analysis of Deep Learning Models for Sentiment Analysis on IMDB Reviews,” *J. Electr. Syst.*, vol. 20, no. 2s, 2024

[7] M. Kayed, R. P. Díaz-Redondo, and A. Mabrouk, “Deep Learning-based Sentiment Classification: A Comparative Survey,” *arXiv preprint* arXiv:2312.17253, Dec. 2023.

[8] G. Nkhata, S. Gauch, U. Anjum, and J. Zhan, “Fine-tuning BERT with Bidirectional LSTM for Fine-grained Movie Reviews Sentiment Analysis,” *arXiv preprint* arXiv:2502.20682, Feb. 2025.

[9] Z. Zhao et al., “Sentiment Analysis Based on Deep Learning: A Comparative Study,” *Electronics*, vol. 9, no. 3, pp. 483–499, Mar. 2020.

[10] R. Garg and A. K. Saha, “An Exhaustive Comparative Study of Machine Learning Algorithms for Natural Language Processing Applications,” *Electronics*, vol. 12, no. 1, pp. 118–134, Jan. 2023.

[11] L. . Damayanti and K. M. . Lhaksmana, “Sentiment Analysis of the 2024 Indonesia Presidential Election on Twitter”, *SinkrOn*, vol. 8, no. 2, pp. 938-946, Mar. 2024.

[12] L. Geni, E. Yulianti, and D. I. Sensuse, “Sentiment Analysis of Tweets Before the 2024 Elections in Indonesia Using Bert Language Models”, *J. Ilm. Tek. Elektro Komput. Dan Inform*, vol. 9, no. 3, pp. 746–757, Aug. 2023.

[13] T. C. Herdiyani and A. U. Zailani, "Sentiment analysis terkait pemindahan Ibu Kota Indonesia menggunakan metode Random Forest berdasarkan tweet warga negara Indonesia," *Jurnal Teknologi Sistem Informasi*, vol. 3, no. 2, pp. 154–165, 2022.

[14] R. Sanusi, F. D. Astuti, and I. Y. Buryadi, "Analisis sentimen pada Twitter terhadap program kartu pra kerja dengan recurrent neural network," *JIKO (Jurnal Informatika dan Komputer)*, vol. 5, no. 2, pp. 89–99, 2021.

[15] T. Sabrila, Y. Azhar, and C. Aditya, "Analisis sentimen tweet tentang UU Cipta Kerja menggunakan algoritma SVM berbasis PSO," *JISKA (Jurnal Informatika Sunan Kalijaga)*, vol. 7, pp. 10–19, 2022, doi: 10.14421/jiska.2022.7.1.10-19.

[16] G. Nkhata, A. K. Bansal, J. H. Oduor, and H. S. Chae, “Performance evaluation of BERT with BiLSTM on IMDB reviews,” *arXiv preprint arXiv:2502.20682*, Feb. 2025.

[17] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, “Learning word vectors for sentiment analysis,” in *Proc. 49th Annu. Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, OR, USA, Jun. 2011, pp. 142–150.