Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 3, Maret 2026
e-ISSN 2685-998X
DOI 10.30865/json.v7i3.9637

Improved Text Classification for Indonesian Hate Speech Detection: FastText-LSTM Model with Easy Data Augmentation

Hilman Singgih Wicaksana[1,*], **Khairul Huda**[1], **Gregorius Airlangga**[2]
[1] Informatics Program, Faculty of Law, Management, and Informatics, Universitas Karya Husada, Semarang, Indonesia
[2] Information Systems Program, Faculty of Bioscience, Technology, and Innovation, Universitas Katolik Indonesia Atma Jaya, Jakarta, Indonesia
Email: [1,*]singgih.hilman@gmail.com, [2]khairulhuda@unkaha.ac.id, [3]gregorius.airlangga@atmajaya.ac.id
Correspondence Author Email: singgih.hilman@gmail.com*
Submitted: **02/03/2026**; Accepted: **26/03/2026**; Published: **31/03/2026**

Abstract− The swift expansion of social media in Indonesia has led to a significant rise in hate speech, highlighting the urgent need for effective automated detection techniques. This research evaluates the performance of the proposed FastText-Long Short-Term Memory with Easy Data Augmentation (FastText-LSTM-WE) compared with the baseline model, FastText-Convolutional Neural Network with Easy Data Augmentation (FastText-CNN-WE). To further investigate the impact of data augmentation, the effectiveness of both FastText-Long Short-Term Memory without Easy Data Augmentation (FastText-LSTM-WO) and FastText-Convolutional Neural Network without Easy Data Augmentation (FastText-CNN-WO) was also assessed. Bayesian Optimization was employed to identify the best hyperparameter configurations for each model. The experiments were carried out on a dataset comprising 14,306 samples while maintaining consistent experimental conditions. Model performance was measured using precision, recall, F1-score, and accuracy derived from the confusion matrix. The results indicate that FastText-LSTM-WE achieved the highest performance, with precision, recall, F1-score, and accuracy of 84.02%, 83.16%, 83.59%, and 81.37%, respectively. These findings demonstrate that the proposed model provides a robust and reliable solution for detecting hate speech within the Indonesian context, thereby improving automated content moderation systems in practical applications.

Keywords: Bayesian Optimization; Easy Data Augmentation; FastText; Hate Speech Detection; Long Short-Term Memory

1. INTRODUCTION

The advancement of information and communication technology has led to a significant increase in social media usage in Indonesia. Digital platforms have become primary spaces for individuals to express opinions, discuss societal issues, and disseminate information rapidly and widely [1]. In this context, hate speech refers to verbal expressions that attack specific groups based on shared characteristics, such as race, gender, ethnicity, religion, or political affiliation [2]. The widespread dissemination of hate speech on social media can trigger social conflict, intensify polarization, and reduce the quality of public discourse in digital environments [3]. Therefore, an automated computational approach is required to accurately and efficiently detect and classify hate speech content.

Traditional rule-based approaches have limitations in identifying hate speech due to its diverse linguistic variations and contextual expressions [4]. Hate speech may appear in different sentence structures while maintaining the same offensive intent toward targeted groups [5], [6]. These variations make pattern-based detection methods less effective, as they rely on predefined rules that cannot capture complex semantic relationships [7]. Consequently, machine learning and deep learning approaches are more suitable, as they enable models to learn patterns from data automatically. Deep learning models, in particular, have demonstrated strong capability in capturing contextual and sequential information in textual data.

In addition to model architecture, text representation is critical for hate speech detection. Word embeddings convert textual data into numerical vectors that capture semantic relationships in a multidimensional space [8]. FastText is a word embedding model that represents words using character-level n-grams, allowing it to generate meaningful representations even for rare or unseen words [9]. This capability is especially relevant in hate speech detection, where offensive expressions often involve spelling variations or modified words. Therefore, FastText embeddings can enhance the quality of textual representation and improve classification performance.

Previous studies have explored various deep learning architectures for hate speech detection. The study in [10] applied LSTM and BiLSTM models on a multilabel hate speech dataset containing 13,169 tweets and reported accuracies of 78.67% and 80.25%, respectively. Similarly, the study in [11] evaluated several architectures, including BERT-CNN, BERT-LSTM, and BERT-BiLSTM, and found that BERT-BiLSTM achieved the highest accuracy of 82% and an F1-score of 81.2%. Another study in [12] utilised a FastText-BiLSTM model combined with oversampling techniques, achieving an F1-score of 75.3% using SMOTE. These findings indicate that recurrent neural network architectures and FastText embeddings are effective for hate speech classification tasks.

However, several limitations remain in prior studies. Most research has focused primarily on architectural variations, without systematically integrating data augmentation techniques to improve the diversity of the training data. Traditional oversampling methods, such as SMOTE, may not consistently improve performance under severe class imbalance [12]. In addition, systematic hyperparameter tuning procedures were often not explicitly conducted, which may limit model optimization and performance stability. These limitations highlight the need for a more comprehensive experimental framework that integrates data augmentation and structured hyperparameter optimization. Such an approach is essential for improving robustness and ensuring fair comparisons across different deep learning architectures.

This research aims to assess the effectiveness of FastText-LSTM, the proposed primary model, compared with FastText-CNN, the baseline model, for identifying hate speech in Indonesian text. Both models are evaluated under the same experimental conditions, using FastText embeddings for text representation, Easy Data Augmentation (EDA) to enhance the diversity of the training data, and Bayesian Optimization for systematic hyperparameter tuning. The experiments are conducted on a dataset comprising 14,306 samples to ensure a comprehensive evaluation. By providing a controlled comparison between recurrent and convolutional architectures, this study aims to identify an efficient and optimized model for detecting hate speech in the Indonesian context. Additionally, this research evaluates whether integrating EDA and Bayesian Optimization can improve the performance and robustness of the FastText-LSTM model. The main contributions of this study are to provide an empirical evaluation of the effectiveness of FastText-LSTM combined with EDA and Bayesian Optimization, and to present a comparative analysis with FastText-CNN under identical experimental settings for Indonesian hate speech detection.

2. RESEARCH METHODOLOGY

2.1 Research Stages

This research involves several essential phases: data preparation, dataset splitting, word embedding, hyperparameter tuning, model evaluation, and model selection. The data preparation phase focuses on cleaning and standardizing the text data to prepare it for model training. Next, the dataset is split into training and testing sets using a cross-validation method to ensure a fair evaluation. During the word embedding phase, FastText is utilized to create numerical vector representations of words based on subwords. The hyperparameter tuning phase uses Bayesian Optimization to identify the parameter combination that maximizes model performance. Afterwards, the model is assessed using metrics such as accuracy, precision, recall, and F1-score to evaluate its effectiveness. Lastly, model selection is performed by comparing the performance of FastText-LSTM and FastText-CNN to identify the most effective model for hate speech detection, as shown in Figure 1.

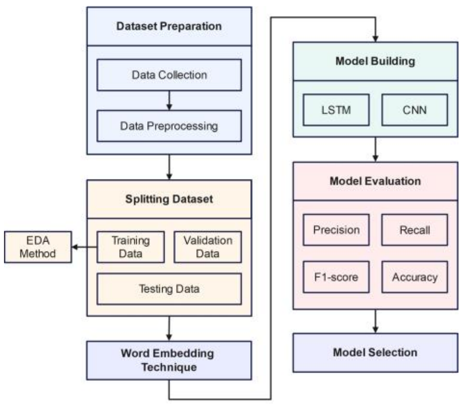


Figure 1. Research Stages

2.2 Dataset Preparation

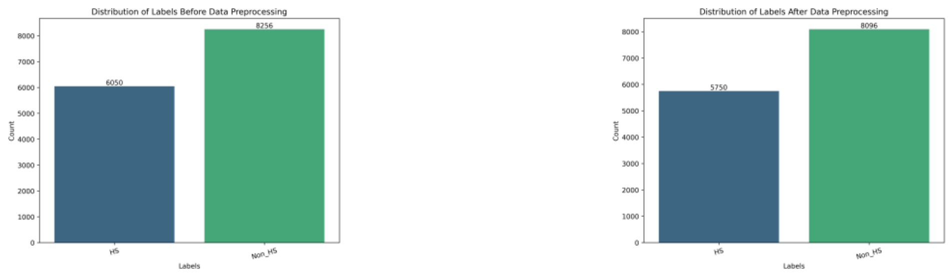
This section provides details on the dataset used in this research, sourced from Tonneau et al. [13]. The dataset used in this study consists of Indonesian social media texts categorized into two distinct groups: hate speech and non-hate speech. In total, the dataset contains 14,306 text samples, including 6,050 instances of hate speech and 8,256 instances of non-hate speech. This dataset was selected for its representation of online discourse in Indonesia and its frequent usage in previous studies on hate speech detection. Its variety of linguistic expressions and contextual differences makes it well-suited for evaluating deep learning models, as illustrated in Figure 2(a).

Table 1. Example of Preprocessing Results

Phases	Before Preprocessing	After Preprocessing
Case Folding	*Kritik dan menghina itu beda cukkk ... @USER*	*kritik dan menghina itu beda cukkk ... @user*
Normalization	*kritik dan menghina itu beda cukkk ... @user*	*kritik menghina beda*
Tokenization	*kritik menghina beda*	['kritik', 'menghina', 'beda']

It's worth noting that this dataset has not been cleaned, so data preprocessing is necessary. These procedures include case folding, normalization, and tokenization. Case folding involves converting all text to lowercase to maintain a consistent representation [14]. Normalization aims to standardize informal expressions, abbreviations, and spelling errors to their correct forms [15]. Tokenization entails dividing sentences into separate words, or tokens, for subsequent analysis. The outcomes of the applied preprocessing techniques are displayed in Table 1.

Once the dataset has been preprocessed, the next step is to ensure that the classes of hate speech and non-hate speech are balanced. After preprocessing, the number of samples in these categories was 5,750 for hate speech and 8,096 for non-hate speech, as shown in Figure 2(b). To tackle this class imbalance, a technique for data augmentation was implemented during the data preparation process to enhance data variety [16]. In particular, the research utilized EDA, a simple yet efficient text augmentation method that creates new training samples through straightforward operations while maintaining the original meaning of the sentences [17], [18], [19]. This involves methods such as synonym replacement, random insertion, random swapping, and random deletion.



(a) (b)

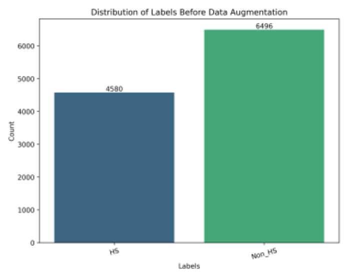
Figure 2. Distribution of Labels: (a) Before Data Preprocessing; (b) After Data Preprocessing

Synonym replacement refers to the practice of replacing certain terms in a sentence with their equivalents while maintaining the overall message [20]. Random insertion involves the unexpected addition of appropriate synonyms into a sentence to improve structural variety [21]. Random swap is the technique of interchangeably changing the positions of two words within a sentence to alter the order of the words [22]. Random deletion consists of eliminating words from sentences based on a certain probability while preserving the primary context [23]. The outcomes of the EDA-based augmentation are presented in Table 2.

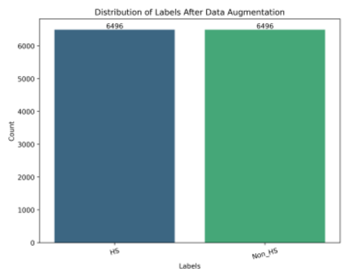
Table 2. Example of Augmentation Results

EDA Technique	Before Augmentation	After Augmentation
Synonym Replacement	*pasti kamu manusia terbuat tanah sudah kecampur **tahi** kucing*	*pasti kamu manusia terbuat tanah sudah kecampur **cirit** kucing*
Random Insertion	*pasti kamu manusia terbuat tanah sudah kecampur tahi kucing*	*pasti kamu manusia terbuat tanah sudah kecampur **insan** tahi kucing*
Random Swap	*pasti kamu manusia terbuat tanah sudah kecampur tahi kucing*	***kecampur** kamu manusia terbuat tanah sudah **pasti** tahi kucing*
Random Deletion	*pasti kamu manusia **terbuat** tanah sudah kecampur tahi kucing*	*pasti kamu manusia tanah sudah kecampur tahi kucing*

According to Table 2, several exploratory data analysis (EDA) techniques were applied to generate augmented text variations. In the synonym replacement technique, the word "*tahi*" in the original sentence "*pasti kamu manusia terbuat tanah sudah kecampur tahi kucing*" was replaced with its synonym "*cirit*", resulting in the modified sentence "*pasti kamu manusia terbuat tanah sudah kecampur cirit kucing*." In the random insertion technique, the word "*insan*" was inserted into the sentence, producing "*pasti kamu manusia terbuat tanah sudah kecampur insan tahi kucing*." In the random swap technique, the position of words was rearranged, transforming the sentence into "*kecampur kamu manusia terbuat tanah sudah pasti tahi kucing*." Lastly, in the random deletion technique, the word "*terbuat*" was removed, resulting in the sentence "*pasti kamu manusia tanah sudah kecampur tahi kucing*." The augmentation process was performed systematically to increase data diversity while preserving the semantic context of the original sentence. Figure 3(a) illustrates the data distribution before augmentation, while Figure 3(b) presents the distribution after augmentation.



(a)



(b)

Figure 4. Distribution of Labels: (a) Before Data Augmentation; (b) After Data Augmentation

2.3 Splitting Dataset

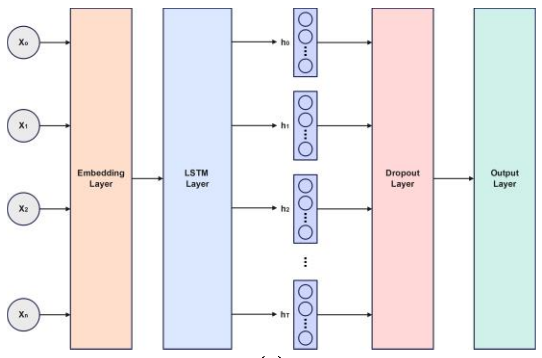
In this research, the dataset was carefully divided into three sections to ensure thorough model training and evaluation. Initially, 80% of the data was allocated for training purposes. This substantial portion provided a robust foundation for the model’s development, allowing it to learn from a diverse array of examples. The next segment, accounting for 10% of the data, was designated as validation data. This subset played a vital role during training, assisting in fine-tuning the model and evaluating its performance on previously unseen data at various stages. Finally, the remaining 10% of the dataset was reserved for testing. This crucial phase was instrumental in assessing the model’s performance after extensive training, as it helped determine its ability to generalize to new data, thereby verifying its reliability and effectiveness for real-world applications.

2.4 Word Embedding Technique

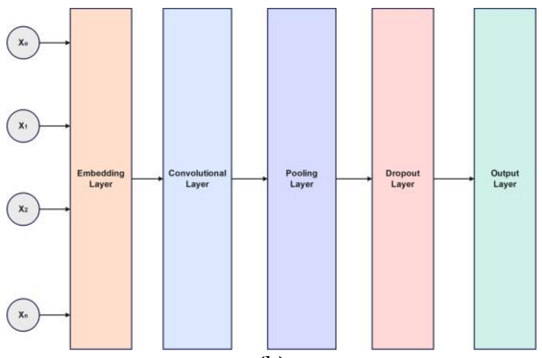
This study employed pretrained FastText embeddings that were trained on the Indonesian Wikipedia corpus using the skip-gram model with a vector dimension of 300 [24]. These pretrained embeddings were used to convert textual data into dense vector representations, effectively capturing both semantic and subword-level information [25], [26]. Initially, without employing EDA, the resulting embedding matrix had a shape of (22691, 300). After applying EDA, the matrix expanded to (22961, 300), indicating improved vocabulary coverage due to the augmented lexical variations, while maintaining the same embedding dimensionality of 300. This expansion highlights the contribution of EDA in enriching the textual representation space utilized for model training.

2.5 Model Building

This research develops two deep learning models: the LSTM model serves as the main model, while the CNN model serves as the reference. The LSTM model’s architecture consists of an input layer, an embedding layer, an LSTM layer, a dropout layer, and an output layer. Figure 5(a) depicts the architecture of the LSTM, providing a visual overview of the data processing sequence. Conversely, the CNN model’s architecture features an input layer, an embedding layer, a convolutional layer, a pooling layer, a dropout layer, and an output layer. Figure 5(b) demonstrates the CNN structure as the benchmark model, emphasizing the architectural differences with the primary model.



(a)



(b)

Figure 5. Model Architecture: (a) LSTM Model; (b) CNN Model

The input layer is structured to accept and numerically represent preprocessed textual data before forwarding it to the network. The embedding layer converts individual tokens into dense vector representations, enabling the model to capture semantic connections between words [27]. In the LSTM framework, the LSTM layer captures long-term dependencies and sequential patterns within the text data [28]. On the other hand, in the CNN framework, the convolutional layer identifies local features using sliding filters [29]. In contrast, the max pooling layer selects the most important feature values to reduce dimensionality while preserving the most informative signals [30]. To mitigate overfitting, the dropout layer randomly drops a specified percentage of neurons during training [31]. Ultimately, the output layer generates the final classification outcome by converting the learned representations into probabilities for the target classes.

Table 3. Values for Each Hyperparameter

Models	Hyperparameters	Values
LSTM	dropout	0.3, 0.5
	learning_rate	0.00001, 0.0001
	lstm_units	64, 128
CNN	dropout	0.3, 0.5
	learning_rate	0.00001, 0.0001
	filters	64, 128
	kernel_size	3, 5

During model development, various hyperparameters were tuned to achieve optimal performance. For the LSTM model, the hyperparameters included dropout rates of 0.3 and 0.5, learning rates of 0.00001 and 0.0001, and LSTM unit sizes of 64 and 128. For the CNN model, the hyperparameters comprised dropout rates of 0.3 and 0.5, learning rates of 0.00001 and 0.0001, and filter sizes of 64 or 128, with kernel sizes of 3 or 5. The chosen ranges for these values aimed to explore an optimal model configuration while avoiding unnecessary complexity. A detailed summary of all combinations of these hyperparameter values is provided in Table 3.

The process of tuning hyperparameters occurs during model training, using both the training and validation datasets. To ensure a more precise and impartial assessment of the model's performance, the K-fold cross-validation method with 5 folds was applied. This approach guarantees that the model exhibits strong generalization abilities when faced with new data that was not part of the training dataset [32]. Furthermore, Bayesian Optimization is used to find the optimal combinations of hyperparameters within a specified search space [33]. This approach improves the optimization process's efficiency and increases the model's ability to achieve its highest performance.

2.6 Model Evaluation

Evaluating a model is a vital process that measures how accurately and consistently machine learning models predict outcomes based on test data [34], [35]. In classification tasks, performance evaluation hinges on the components of the confusion matrix: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). These elements form the basis for calculating various evaluation metrics that assess the model's performance. In this research, the evaluation was conducted methodically to ensure objective, quantifiable results. This methodology enables a quantitative assessment of model quality by analyzing predictions on test data; the confusion matrix used in this evaluation is depicted in Figure 6.

The metrics used in this research include precision, recall, F1-score, and accuracy, which collectively provide a comprehensive evaluation of performance. Precision assesses the accuracy of positive predictions by comparing the count of true positive predictions to the total number of positive predictions made by the model [36]. On the other hand, recall measures the model’s capability to detect all actual positive instances [37]. The F1-score represents the harmonic mean of precision and recall, providing a balance between these two metrics, particularly in cases with skewed class distributions [38]. Accuracy is defined as the proportion of correct predictions relative to the total number of test data points [39]. The mathematical definitions for precision, recall, F1-score, and accuracy are presented in Formulas (1) to (4), respectively.

$$Precision\ (P) = \frac{TP}{TP+FP} \quad (1)$$

$$Recall\ (R) = \frac{TP}{TP+FN} \quad (2)$$

$$F1-score = 2 \times \frac{P \times R}{P+R} \quad (3)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

3. RESULT AND DISCUSSION

This section consists of three key components: model evaluation, model selection, and a comparative analysis with prior studies. Model evaluation assesses the model's performance, enabling monitoring of the learning process and its effectiveness, especially its ability to generalize to unseen data. In parallel, model selection specifies the criteria for selecting the best-performing model based on specific evaluation metrics, including accuracy, precision, recall, and F1-score. After the model selection process, a comparative analysis with existing studies is presented to demonstrate how the proposed approach compares with established hate speech detection methods. Collectively, these analyses offer a thorough perspective on the effectiveness and contributions of the proposed model.

3.1 Model Comparison Analysis

This section assesses the effectiveness of each model developed in this study using test data. This evaluation aims to assess how well each model classifies unseen data that were not part of the training process. We examine model performance through various metrics, including precision, recall, and F1-score, and accuracy, all of which are based on the elements of the confusion matrix: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). These metrics provide a thorough insight into prediction accuracy, the ability to identify positive categories, and the overall equilibrium of model performance. The full confusion matrix, along with the assessment metrics for each model, is provided in Table 4.

Each model was assessed using the hyperparameter combination identified as most effective by Bayesian Optimization during the training phase. The FastText-LSTM-WE model achieved optimal performance with a dropout rate of 0.3, a learning rate of 0.001, and 32 LSTM units, while the FastText-LSTM-WO model operated with a dropout rate of 0.5, a learning rate of 0.0001, and also 32 LSTM units. For the CNN architecture, the FastText-CNN-WE model employed a dropout rate of 0.5, 64 filters, a kernel size of 3, and a learning rate of 0.001. In contrast, the FastText-CNN-WO model utilized a dropout rate of 0.5, 64 filters, a kernel size of 5, and a learning rate of 0.0001. These optimized hyperparameter configurations were used to train the final models, which were then evaluated on the test dataset. This methodology ensures that each model is assessed under the most favourable conditions established during hyperparameter tuning.

Table 4. Performance Comparison of Each Model

Models	TP	FN	FP	TN	Evaluation Metrics			
					Precision	Recall	F1-score	Accuracy
FastText-LSTM-WE	**657**	**133**	**125**	**470**	**0.8402**	**0.8316**	**0.8359**	**0.8137**
FastText-LSTM-WO	627	159	163	436	0.7977	0.7937	0.7957	0.7675
FastText-CNN-WE	651	132	139	463	0.8314	0.8241	0.8277	0.8043
FastText-CNN-WO	635	170	155	425	0.7888	0.8038	0.7962	0.7653

The results from the confusion matrix, as shown in Table 4, were utilized to calculate various evaluation metrics using the formulas outlined in the research methodology. These metrics include precision, recall, F1-score, and accuracy, which are standard measures of classification model performance. The calculations were based on the TP, TN, FP, and FN values extracted from the confusion matrix. This step ensures that the evaluation results align with the performance measurement approach described in the methodology section. To illustrate this process, the manual calculations of the evaluation metrics for the FastText-LSTM-WE model are provided below.

Precision is determined by Equation (1), using the confusion matrix values.

$$Precision = \frac{657}{657 + 125}$$

$$Precision = \frac{657}{782}$$

$$Precision = 0.8402$$

Recall is calculated using Equation (2), with the confusion matrix values substituted.

$$Recall = \frac{657}{657 + 133}$$

$$Recall = \frac{657}{790}$$

$$Recall = 0.8316$$

F1-score is determined using Equation (3), by substituting the calculated values.

$$F1 - score = \frac{2 \times 0.8316 \times 0.8402}{0.8316 + 0.8402}$$

$$F1 - score = \frac{2 \times 0.6987}{1.6718}$$

$$F1 - score = \frac{1.3974}{1.6718}$$

$$F1 - score = 0.8359$$

Accuracy is calculated using Equation (4) by substituting the confusion matrix values.

$$Accuracy = \frac{657 + 470}{657 + 470 + 125 + 133}$$

$$Accuracy = \frac{1127}{1385}$$

$$Accuracy = 0.8137$$

The experimental results indicate that the FastText-LSTM-WE model achieved the highest overall performance among the evaluated models, with a precision of 0.8402 (84.02%), recall of 0.8316 (83.16%), F1-score of 0.8359 (83.59%), and accuracy of 0.8137 (81.37%). Compared with the FastText-LSTM-WO model, the EDA implementation resulted in significant improvements across all evaluation metrics, including increases in precision from 0.7977 (79.77%) to 0.8402 (84.02%) and recall from 0.7937 (79.37%) to 0.8316 (83.16%). The F1-score also improved, moving from 0.7957 (79.57%) to 0.8359 (83.59%), while accuracy enhanced from 0.7675 (76.75%) to 0.8137 (81.37%). A similar trend was observed with the CNN architecture, where the FastText-CNN-WE model outperformed the FastText-CNN-WO model, achieving a precision of 0.8314 (83.14%) and an accuracy of 0.8043 (80.43%). Overall, these findings suggest that applying EDA significantly enhances the classification performance of both LSTM and CNN models.

3.2 Model Selection

In this section, the optimal model is identified based on its performance on the test dataset using several evaluation metrics, including precision, recall, F1-score, and accuracy. These metrics provide a comprehensive assessment of the model's ability to distinguish between hate speech and non-hate speech. The selection process emphasizes the importance of achieving balanced performance across these metrics, as relying on a single metric may not fully reflect the model's effectiveness, especially with imbalanced data distributions. Consequently, the evaluation considers how well each model maintains consistency between precision and recall while also achieving high overall accuracy. This holistic approach ensures a more robust classification capability within the task.

Based on the experimental results, the FastText-LSTM-WE model showed the best overall performance among all evaluated models. It achieved 81.37% accuracy and 83.59% F1-score, indicating a solid balance between precision and recall in detecting hate speech. Additionally, it recorded a precision of 84.02% and a recall of 83.16%, demonstrating its effectiveness in accurately identifying hate speech while reducing errors. These results suggest that integrating FastText embeddings, EDA, and Bayesian Optimization greatly enhances the model's robustness and effectiveness. Consequently, the FastText-LSTM-WE model has been selected as the best option and will serve as the primary method for hate speech detection in Indonesian text in this study.

3.3 Comparative Analysis with Previous Studies

To evaluate the effectiveness of the proposed approach, the results from this study were compared with those of several previous studies on Indonesian hate speech detection. A study conducted by Zahra et al. [10] applied LSTM and BiLSTM models on an Indonesian Twitter dataset and reported accuracies of 78.67% and 80.25%, respectively. Another study by Saputra et al. [11] implemented a hybrid deep learning architecture based on BERT-BiLSTM and achieved an accuracy of 82% with an F1-score of 81.2%. In addition, Faza et al. [12] utilized a FastText-BiLSTM model combined with SMOTE oversampling techniques and obtained an F1-score of 75.3%. Compared with these studies, the proposed FastText-LSTM-WE model achieved an F1-score of 83.59% and an accuracy of 81.37%, indicating competitive performance in Indonesian hate speech detection tasks. The improved performance can be attributed to the integration of EDA, which increases the diversity of the training data, and to the use of Bayesian Optimization to systematically determine optimal hyperparameter configurations. A detailed comparison of the results with previous studies is presented in Table 5.

Table 5. Performance Comparison of Each Model

Study	Method	F1-score	Accuracy
Zahra et al. [10]	LSTM	–	78.67%
Zahra et al. [10]	BiLSTM	–	80.25%
Saputra et al. [11]	BERT-BiLSTM	81.20%	82.00%
Faza et al. [12]	FastText-BiLSTM + SMOTE		
Proposed Model	**FastText-LSTM + EDA + Bayesian Optimization**	**83.59%**	**81.37%**

The results presented in Table 5 indicate that the proposed FastText-LSTM-WE model achieved an accuracy of 81.37% and an F1-score of 83.59%, which are competitive with several previous studies on hate speech detection in Indonesian. For example, research by Zahra et al. [10] employed LSTM and BiLSTM models with accuracies of 78.67% and 80.25%, respectively, while Saputra et al. [11] used a BERT-BiLSTM architecture, achieving an accuracy of 82% and an F1-score of 81.2%. Although the BERT-BiLSTM model achieves slightly higher accuracy, it relies on a transformer architecture that is computationally more complex. Additionally, Faza et al. [12] combined FastText-BiLSTM with SMOTE oversampling, achieving an F1-score of 75.3%. While our study has limitations, as it only utilizes LSTM and CNN architectures without more complex transformer models, it contributes significantly by integrating FastText embeddings, EDA, and Bayesian Optimization within a single controlled experimental framework to enhance model performance. This approach demonstrates that improved classification performance can be achieved not only by adopting more complex architectures but also by increasing the diversity of training data and systematically optimizing hyperparameters, offering a more efficient alternative for hate speech detection in Indonesian-language text.

4. CONCLUSION

This research focused on developing and evaluating hate speech detection models using FastText embeddings with LSTM and CNN architectures, with and without EDA. The experimental results indicated that EDA consistently enhanced model performance across both architectures, as evidenced by improvements in accuracy, precision, recall, and F1-score compared to models trained without augmentation. Among the models evaluated, FastText-LSTM-WE demonstrated the highest performance on the test dataset, achieving a precision of 84.02%, a recall of 83.16%, an F1-score of 83.59%, and an accuracy of 81.37%, underscoring its effectiveness in differentiating between hate speech and non-hate speech. However, these performance gains should be interpreted within the context of the experimental setup, which involved a specific dataset, predefined preprocessing techniques, and controlled hyperparameter configurations. Additionally, Bayesian Optimization played a crucial role in identifying optimal hyperparameter settings, further enhancing overall model performance. Nonetheless, several limitations must be acknowledged, including the focus on LSTM and CNN architectures without exploring more advanced transformer-based models, the dataset's limited representativeness across various real-world scenarios, and the use of relatively basic augmentation techniques. Consequently, future research is encouraged to investigate transformer-based or hybrid architectures, implement more advanced data augmentation strategies, such as contextual or generative approaches, and assess model performance on larger, more diverse datasets to improve generalizability and robustness in real-world hate speech detection applications.

REFERENCES

[1] A. Nurdin, A. N. Paryati, S. K. Rizqi, I. H. Hermawan, and T. Q. Handayani, “The Role of Social Media in Political Education and Election Socialization Among Generation Z,” *The Journal of Academic Science*, no. 2, pp. 566–577, 2025, doi: https://doi.org/10.59613/2w7p1883.

[2] A. Dreißigacker, P. Müller, A. Isenhardt, and J. Schemmel, “Online hate speech victimization: consequences for victims’ feelings of insecurity,” *Crime Sci.*, vol. 13, no. 1, Dec. 2024, doi: 10.1186/s40163-024-00204-y.

[3] H. Berchenko, P. Domingos, D. Shahiqi, Z. Fetahu, and R. Fetahu, “The Criminal Confrontation for the Crimes of Discrimination and Hate Speech: A Comparative Study,” *Access to Justice in Eastern Europe*, vol. 7, no. 2, pp. 138–162, 2024, doi: https://doi.org/10.33327/AJEE-18-7.2-000210.

[4] P. Ray and A. Chakrabarti, “A Mixed approach of Deep Learning method and Rule-Based method to improve Aspect Level Sentiment Analysis,” *Applied Computing and Informatics*, vol. 18, no. 1–2, pp. 163–178, Jan. 2022, doi: 10.1016/j.aci.2019.02.002.

[5] M. Vergani *et al.*, “Mapping the scientific knowledge and approaches to defining and measuring hate crime, hate speech, and hate incidents: A systematic review,” *Campbell Systematic Reviews*, vol. 20, no. 2, pp. 1–54, Jun. 2024, doi: 10.1002/cl2.1397.

[6] J. Zapata and O. Deroy, “Ordinary citizens are more severe towards verbal than nonverbal hate-motivated incidents with identical consequences,” *Sci. Rep.*, vol. 13, no. 1, Dec. 2023, doi: 10.1038/s41598-023-33892-8.

[7] J. M. Perez *et al.*, “Assessing the Impact of Contextual Information in Hate Speech Detection,” *IEEE Access*, vol. 11, pp. 30575–30590, 2023, doi: 10.1109/ACCESS.2023.3258973.

[8] P. Poschmann, J. Goldenstein, S. Büchel, and U. Hahn, “A Vector Space Approach for Measuring Relationality and Multidimensionality of Meaning in Large Text Collections,” *Organ. Res. Methods*, vol. 27, no. 4, pp. 650–680, Oct. 2024, doi: 10.1177/10944281231213068.

[9] O. Karakaya and Z. H. Kilimci, “An efficient consolidation of word embedding and deep learning techniques for classifying anticancer peptides: FastText+BiLSTM,” *PeerJ Comput. Sci.*, vol. 10, 2024, doi: 10.7717/peerj-cs.1831.

[10] E. Aurora Az Zahra, Y. Sibaroni, and S. Suryani Prasetyowati, “Classification of Multi-Label of Hate Speech on Twitter Indonesia using LSTM and BiLSTM Method,” *JINAV: Journal of Information and Visualization*, vol. 4, no. 2, pp. 170–178, Jul. 2023, doi: 10.35877/454ri.jinav1864.

[11] R. Angger Saputra and Y. Sibaroni, "Multilabel Hate Speech Classification in Indonesian Political Discourse on X using Combined Deep Learning Models with Considering Sentence Length," *Jurnal Ilmu Komputer dan Informasi*, vol. 18, no. 1, pp. 113–125, Feb. 2025, doi: 10.21609/jiki.v18i1.1440.

[12] A. Muhamad Faza, Y. Sibaroni, and S. S. Prasetyowati, "A Comparative Study on Handling Imbalanced Data in Indonesian Hate Speech Detection Using FastText and BiLSTM," *Intl. Journal on ICT*, vol. 11, no. 2, pp. 136–149, 2025, doi: 10.21108/ijoict.v11i2.9513.

[13] M. Tonneau, D. Liu, S. Fraiberger, R. Schroeder, S. Hale, and P. Röttger, "From Languages to Geographies: Towards Evaluating Cultural Bias in Hate Speech Datasets," in *Proceedings of the 8th Workshop on Online Abuse and Harms (WOAH 2024)*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2024, pp. 283–311. doi: 10.18653/v1/2024.woah-1.23.

[14] A. T. Ni'mah and R. Yunitarini, "Relevance of the Retrieval of Hadith Information (RoHI) using Bidirectional Encoder Representations from Transformers (BERT) in religious education media," in *BIO Web of Conferences*, EDP Sciences, Nov. 2024. doi: 10.1051/bioconf/202414601041.

[15] A. Ahmad Aliero, B. Sulaimon Adebayo, H. Olanrewaju Aliyu, A. Gogo Tafida, B. Umar Kangiwa, and N. Muhammad Dankolo, "Systematic Review on Text Normalization Techniques and its Approach to Non-Standard Words," *Int. J. Comput. Appl.*, vol. 185, no. 33, pp. 975–8887, 2023.

[16] V. Maslej-Krešňáková, M. Sarnovský, and J. Jacková, "Use of Data Augmentation Techniques in Detection of Antisocial Behavior Using Deep Learning Methods," *Future Internet*, vol. 14, no. 9, Sep. 2022, doi: 10.3390/fi14090260.

[17] J. Wei and K. Zou, "EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China: Association for Computational Linguistics, 2019, pp. 6382–6388. doi: 10.18653/v1/D19-1670.

[18] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Global Transitions Proceedings*, vol. 3, no. 1, pp. 91–99, Jun. 2022, doi: 10.1016/j.gltp.2022.04.020.

[19] K. Wahyu Trisna, J. Huang, Y. Chen, and I. Gede Juliana Eka Putra, "Dynamic Text Augmentation for Robust Sentiment Analysis: Enhancing Model Performance with EDA and Multi-Channel CNN," *IEEE Access*, vol. 13, pp. 31978–31991, 2025, doi: 10.1109/ACCESS.2025.3538621.

[20] A. R. Nair, R. P. Singh, D. Gupta, and P. Kumar, "Evaluating the Impact of Text Data Augmentation on Text Classification Tasks using DistilBERT," in *Procedia Computer Science*, Elsevier B.V., 2024, pp. 102–111. doi: 10.1016/j.procs.2024.04.013.

[21] S. Latif, N. U. Islam, Z. Uddin, K. M. Cheema, S. S. Ahmed, and M. F. Khan, "Deep ensemble learning with transformer models for enhanced Alzheimer's disease detection," *Sci. Rep.*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-08362-y.

[22] S. Wang and Y. Xiang, "Research on Data Augmentation Techniques for Text Classification Based on Antonym Replacement and Random Swapping," in *Proceedings of the International Conference on Modeling, Natural Language Processing and Machine Learning*, New York, NY, USA: ACM, May 2024, pp. 103–108. doi: 10.1145/3677779.3677796.

[23] A. S. Moon, K. Kim, and J. Lee, "Data augmentation for dense passage retrieval using corpus-passage frequency-based token deletion," *J. Big Data*, vol. 12, no. 1, Dec. 2025, doi: 10.1186/s40537-025-01257-9.

[24] A. Reyhandro and A. Faza, "Detecting Evergreen Articles: Comparing Word Embeddings for Hybrid Bidirectional GRU and LSTM Model on Indonesian News Portal," in *2025 5th International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS)*, IEEE, Aug. 2025, pp. 78–83. doi: 10.1109/ICE3IS66769.2025.11281212.

[25] P. Mojumder, M. Hasan, Md. F. Hossain, and K. M. Hasan, "A Study of fastText Word Embedding Effects in Document Classification in Bangla Language," in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (LNICST)*, vol. 325, Springer, Cham, 2020, pp. 441–453. doi: 10.1007/978-3-030-52856-0_35.

[26] U. S. Yadavalli and S. R. Sahoo, "A multi-granular hybrid neural architecture for detecting abusive content in online social networks (OSNs) with contextual awareness," *J. Big Data*, vol. 13, no. 1, Dec. 2025, doi: 10.1186/s40537-025-01343-y.

[27] P. Kumari, A. Thakur, G. Kaur, N. Yadav, B. Singh, and E. Uchoi, "Text-based Misinformation Detection with Sequential and Convolutional Neural Networks," in *2025 IEEE 7th International Conference on Computing, Communication and Automation (ICCCA)*, IEEE, Nov. 2025, pp. 1–6. doi: 10.1109/ICCCA66364.2025.11325151.

[28] M. Krichen and A. Mihoub, "Long Short-Term Memory Networks: A Comprehensive Survey," Sep. 01, 2025, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/ai6090215.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 3, Maret 2026
e-ISSN 2685-998X
DOI 10.30865/json.v7i3.9637

[29] Y. Zhang, Y. Ma, and Y. Liu, “Convolution-Bidirectional Temporal Convolutional Network for Protein Secondary Structure Prediction,” *IEEE Access*, vol. 10, pp. 117469–117476, 2022, doi: 10.1109/ACCESS.2022.3219490.

[30] L. Zhao and Z. Zhang, “A improved pooling method for convolutional neural networks,” *Sci. Rep.*, vol. 14, no. 1, Dec. 2024, doi: 10.1038/s41598-024-51258-6.

[31] K. K. D. Surendranath, M. Sam, and L. Qingge, “An Adaptive Dropout and Parallel Computing Approaches for Accelerating RNN Controller,” in *2024 IEEE 15th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2024*, Institute of Electrical and Electronics Engineers Inc., 2024, pp. 667–675. doi: 10.1109/UEMCON62879.2024.10754702.

[32] P. Heidari and A. Milan, “Combining K-fold cross validation with bayesian hyperparameter optimization for accuracy enhancement of land cover and land use classification,” *Sci. Rep.*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-23336-w.

[33] H. S. Wicaksana, R. Kusumaningrum, and R. Gernowo, “Determining community happiness index with transformers and attention-based deep learning,” *IAES International Journal of Artificial Intelligence*, vol. 13, no. 2, pp. 1753–1761, Jun. 2024, doi: 10.11591/ijai.v13.i2.pp1753-1761.

[34] J. H. Cabot and E. G. Ross, “Evaluating prediction model performance,” *Surgery (United States)*, vol. 174, no. 3, pp. 723–726, Sep. 2023, doi: 10.1016/j.surg.2023.05.023.

[35] R. Rein, “Model Evaluation in Machine Learning Applications,” in *Artificial Intelligence and Machine Learning in Sports Science*, D. Memmert, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2025, pp. 41–54. doi: 10.1007/978-3-662-70155-3_3.

[36] S. Sharma and K. Guleria, “A Deep Learning based model for the Detection of Pneumonia from Chest X-Ray Images using VGG-16 and Neural Networks,” in *Procedia Computer Science*, Elsevier B.V., 2022, pp. 357–366. doi: 10.1016/j.procs.2023.01.018.

[37] E. Casas, L. Ramos, E. Bendek, and F. Rivas-Echeverria, “Assessing the Effectiveness of YOLO Architectures for Smoke and Wildfire Detection,” *IEEE Access*, vol. 11, pp. 96554–96583, 2023, doi: 10.1109/ACCESS.2023.3312217.

[38] S. Riyanto, I. S. Sitanggang, T. Djatna, and T. D. Atikah, “Comparative Analysis using Various Performance Metrics in Imbalanced Data for Multi-class Text Classification,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 6, pp. 1082–1090, 2023, doi: 10.14569/IJACSA.2023.01406116PDF.

[39] D. Chicco and G. Jurman, “The Matthews correlation coefficient (MCC) should replace the ROC AUC as the standard metric for assessing binary classification,” *BioData Min.*, vol. 16, no. 1, Dec. 2023, doi: 10.1186/s13040-023-00322-4.