Improving Accuracy with Hyperparameter Tuning for Sarcasm Detection in Twitter Comments using BiLSTM

Enrisa Alyaa Budisantoso, Gumgum Darmawan, and Anindya Apriliyanti Pravitasari

***Abstract*—Sarcasm, a linguistic phenomenon where negative feelings are expressed through positive literal meanings and vice versa, is often used as a coping mechanism or subtle commentary on hardships. While sarcasm is a common way to express negativity through seemingly positive statements, its detection in Indonesian social media, particularly regarding user irritation with slow internet connectivity, remains challenging. This paper tackles the challenge of detecting sarcasm in imbalanced datasets by the utilization of Bidirectional Long Short-Term Memory (BiLSTM) model with optimized hyperparameters and oversampling techniques. The dataset exhibited a significant class imbalance. Oversampling is necessary, with Random Over Sampling (ROS) being the most effective for handling sarcasm. Optimal hyperparameters for the BiLSTM model are determined, including 128 neurons in the BiLSTM layer, Adam optimizer, dropout rate of 0.3, batch size of 16, 32 neurons in the dense layer, and 0.0001 as the learning rate, resulting in excellent F1-score, accuracy, precision, and recall for sarcastic and non-sarcastic instances. The selected hyperparameter configuration achieves 97.40% accuracy, demonstrating its effectiveness in predicting both classes. This research improves the comprehension of sarcasm identification in Indonesian social media and presents potential for sentiment analysis in the digital space.**

***Index Terms*—BiLSTM, imbalanced data, sarcasm, text classification**

I. INTRODUCTION

THE widespread internet adoption in Indonesia reached remarkable heights, with 215.63 million users recorded in 2022–2023, constituting approximately 78.19% of the total population [1]. However, despite this exponential growth, challenges persist in providing equitable and quality internet connectivity, as evidenced by Indonesia's rank of 120 out of 180 countries in terms of internet download speeds [2]. The rise of digital environments has led to a greater prevalence of sarcasm on online networks, where individuals resort to expressing their frustration [3], [4], particularly in response to the challenges posed by low internet speeds in the country.

Sarcasm, a form of nonliteral and figurative language, enables individuals to convey negative emotions through words that have positive literal meanings, and conversely [5]. This linguistic phenomenon often arises as a coping mechanism or a subtle expression in response to challenges [6], such as the frustration induced by slow internet speeds in Indonesia. While using sarcasm on social media may serve to navigate platform guidelines subtly, interpreting sarcasm remains a complex task due to its inherent nuances and context-dependent nature [7].

In the context of business in Indonesia, companies have been focusing on analyzing customer sentiment towards products. Recognizing that sarcasm can cause bias in sentiment analysis, it is crucial to analyze sarcasm detection carefully. This step is taken to obtain more accurate sentiment analysis results and provide deeper insights into customer perceptions of products. By doing so, companies can respond more appropriately to customer feedback, improve product quality, and strengthen consumer relationships.

Researchers have conducted several studies using machine learning methods in text analysis. The methods include Support Vector Machine (SVM) [8], [9], Naïve Bayes method [8], [10], Decision Tree method [10], Random Forest method [11], and Bidirectional Long Short-Term (BiLSTM) method [5], [7], [12].

Previous research commonly employed various feature extractors, including TF–IDF, Word2Vec, and BoW, along with conventional machine learning algorithms for sentiment classification. Nevertheless, these models frequently encountered obstacles, including low accuracy as a result of imbalanced class distribution, inaccurate preprocessing techniques, and high-dimensional feature vectors [13]. BiLSTM, especially with attention mechanism, emerges as a prominent method for addressing the complexity of sarcasm in intricate sentence structures and ambiguous meanings [14]. In order to comprehend text representations, BiLSTM implements a bidirectional LSTM network. Considering forward and backward directions, the model's understanding of textual structures and meanings is improving [15], [16].

Previous studies have highlighted the efficacy of BiLSTM in text classification tasks, particularly in handling large volumes of textual data. Notable research by [17] demonstrated that BiLSTM outperformed other deep learning algorithms, including LSTM, in textual data classification, achieving an accuracy of 92%. Similarly, the study by [18] comparing LSTM and BiLSTM for sarcasm detection found that BiLSTM exhibited superior performance with an accuracy of 82.55%, surpassing the 81.90% accuracy achieved by standard LSTM. Given the consistent support for the effectiveness of BiLSTM in text classification with higher accuracy, this study adopts the BiLSTM method for its

Manuscript received May 2, 2024; revised May 15, 2025.
This work was supported by Padjadjaran University.
Enrisa Alyaa Budisantoso is a postgraduate student in the Department of Statistics, Padjadjaran University, Indonesia. (email: enrisa20001@mail.unpad.ac.id) Gumgum Darmawan is a lecturer in the Department of Statistics, Padjadjaran University, Indonesia. (email: gumgum@unpad.ac.id)
Anindya Apriliyanti Pravitasari is a lecturer in the Department of Statistics, Padjadjaran University, Indonesia. (email: anindya.apriliyanti@unpad.ac.id)

analysis.

The acquired sarcasm data from Indonesian social media displays an unequal distribution of documents among the different labels. This imbalance introduces the risk of the BiLSTM model inadequately learning both the "sarcastic" and "non-sarcastic " labels. Therefore, this research aims to address the data imbalance to improve the performance of the BiLSTM framework. Three techniques, such as ROS (Random Over Sampling), ADASYN (Adaptive Synthetic Sampling), and SMOTE (Synthetic Minority Over-sampling Technique), are utilized to address the imbalance. Subsequently, a comparative analysis will be performed to ascertain the most effective balancing method.

This study aims to enhance sarcasm detection by employing three oversampling techniques (SMOTE, ADASYN, and ROS) and tuning hyperparameters in the BiLSTM model. By improving the BiLSTM architecture, the research seeks to deepen the understanding of sarcasm in texts and its applications in communication and sentiment analysis. The goal is to overcome challenges related to literal and hidden meanings in the text, ultimately improving sentiment analysis accuracy in the Indonesian digital landscape.

The organization of the paper is as follows: Section II outlines the materials and methods, Section III presents the experimental results and analysis, Section IV contains the discussion, and the final section concludes the paper.

II. Methodology of Research

The methods and dataset utilized in this research are comprehensively detailed in this section. As follows, the dataset, methods, and analytical workflow are all introduced.

2.1. Data Collection

The foundational dataset for this investigation originated from comments from customers regarding Indonesia's most prominent internet service providers on Twitter, including Indihome, MNC Play, First Media, Biznet, and MyRepublic. The observational timeframe spanned from July 1, 2023, to July 31, 2023, generating 17,768 comments, which constituted the primary focus of analysis in this study. This data collection was done with the help of a Ripple10 dashboard that can only be accessed by certain companies.

The data labeling process for classifying comments into “sarcasm” and “non-sarcasm” categories was executed manually. Comments exhibiting elements of sarcasm were designated with the label (1), while those devoid of sarcasm received the label (0).

2.2. Text Preprocessing

Text preprocessing was an essential initial stage to make textual data comprehensible and effectively processed by text mining systems. This process, as emphasized by [19], was instrumental in influencing the quality of analytical outcomes. The objective of preprocessing was to convert textual data that is unstructured into a more structured format, ensuring that the resulting text information was of high quality and ready for subsequent stages [20]. The process (see Fig. 1) included cleaning, which involved the removal of characters to reducing noise [21]. Case folding was subsequently utilized to transform characters into lowercase, thereby standardizing the writing and improving the accuracy of text classification [22]. Tokenizing divides the text into its smallest components, words, in order to ready the data for examination [23]. Spelling normalization entails the correction or substitution of words containing spelling errors or particular abbreviations within text data, whereas stopword removal involves the elimination of less significant words to emphasize more informative terms [24]. Stemming is a procedure that eliminates affixes and organizes phrases with a common base form, and increases the number of documents retrieved through search and indexing systems [25]. Text classification requires transforming unstructured natural language into numerical values [26]. The subsequent phase involved word embedding, in which words were represented as vectors to represent their semantic and syntactic meanings [27], given its exceptional performance in classifying text data, FastText was selected for this study.

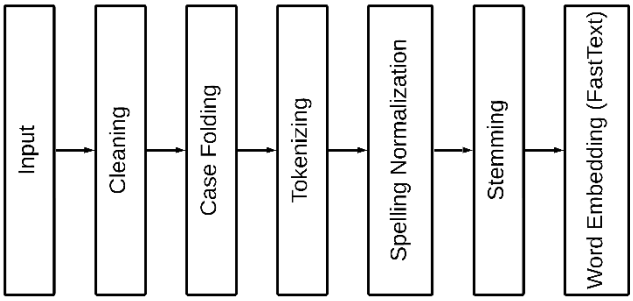


Fig. 1. Text Preprocessing Steps

2.3. Imbalanced Classification

Imbalanced class pertained to a dataset characterized by an uneven distribution of class frequencies, where the category with the highest occurrence was denoted as the majority, and the one with a comparatively lower occurrence was labeled as the minority [28]. This disparity in class proportions posed inherent challenges, as machine learning algorithms primarily acquire information from the majority class, potentially compromising their performance, especially in the context of minority classes [29]. The repercussions of such imbalances manifested in a decline in the model's effectiveness in precisely identifying and categorizing samples from the minority class [30]. Table I below illustrates the extent of the imbalance in the proportion of minority classes, according to Google for Developers [31].

TABLE I
DEGREE OF IMBALANCE

Degree of Imbalance	Proportion of Minority Class
Mild	20–40% of the data set
Moderate	1–20% of the data set
Extreme	< 1% of the data set

Consequently, addressing this issue became imperative for robust model development. These studies adopted a comprehensive approach to address the intricacies associated with imbalanced data by incorporating three distinct oversampling techniques: ROS, SMOTE, and ADASYN.

2.3.1. Random Over Sampling (ROS)

The approach of ROS involves iterating through specific cases and randomly introducing additional observations into

the minority class. However, a notable drawback of this method is its potential to induce overfitting by replicating minority instances without intelligent consideration [32]. In Fig. 2, the majority and minority classes are denoted by black and gray bars, respectively. Strategy is grounded in a simple application of the distribution hypothesis, assuming that semantically equivalent terms can be predicted to exist in related documents, such as texts from the minority class.

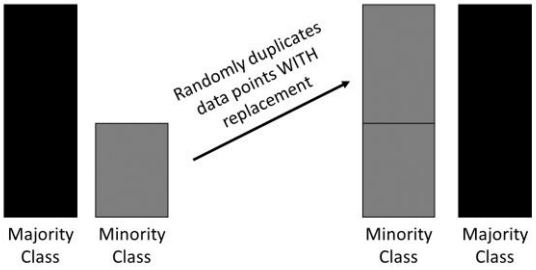


Fig. 2. Illustration of Random Over-Sampling (modified from [33])

2.3.2. SMOTE (Synthetic Minority Over-sampling Technique)

SMOTE is an oversampling method that generates synthetic observations between neighboring minority instances. A key advantage of SMOTE over the ROS method is its ability to address the overfitting issue in machine classification models. SMOTE achieves this by creating synthetic points along straight lines between locations, randomly determining feature vectors and their nearest neighbors in the minority class, and generating supplementary data points through the multiplication of random values within the range of 0 to 1. The synthetic data is incorporated into the training set to improve model training. However, there are two significant limitations associated with this method: its inefficiency in managing high-dimensional data and tends to produce synthetic data without taking into account examples from other classes, potentially leading to more noisy data due to class overlap.

Fig. 3 below illustrates the SMOTE method, where white circles represent the minority (positive) classes and gray circles depict the majority classes. SMOTE creates a synthetic positive sample within the minority class, represented by black circles between white circles. The combination of original and generated positive samples forms the minority group, bringing the majority and minority classes into closer proximity.

Fig. 3. Illustration of SMOTE Oversampling (modified from [34])

2.3.3. ADASYN (Adaptive Synthetic Sampling)

An alternative approach to synthetic data sampling is the ADASYN method. Its primary aim is to achieve a balanced distribution of data categories by dynamically creating samples within the minority class, guided by a specified level of balance in the distribution. The algorithm has two primary objectives: firstly, determining the necessary number of samples for every instance within the minority class, and secondly, prompting algorithms for machine learning to identify or understand difficult instances. The utilization of the ADASYN approach enhances classification performance by mitigating the bias introduced by the original imbalanced dataset, as depicted in Fig. 4. Furthermore, as ADASYN increases the level of balance, there is a noticeable trend toward reducing errors.

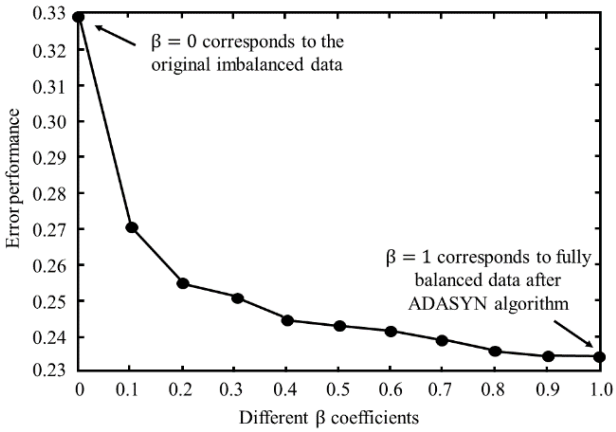


Fig. 4. ADASYN algorithm for different β coefficients (modified from [35])

2.4. Bidirectional Long Short-Term Memory (BiLSTM)

LSTM was developed to address the issue of vanishing gradients, utilizing forget gates, input gates, and output gates to regulate information retention and elimination. While LSTM involves complex calculations and high computation, this study explores an alternative method, BiLSTM, with simpler computation yet comparable performance. The architecture of LSTM is provided in Fig. 5.

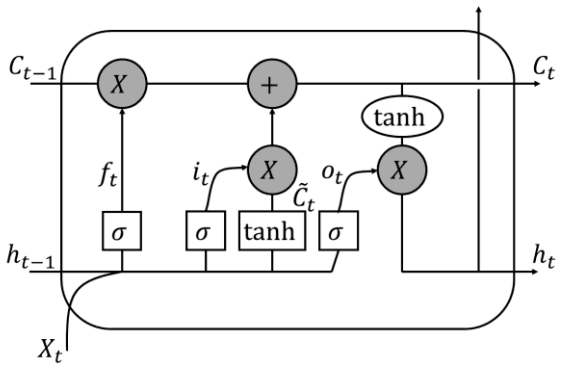


Fig. 5. LSTM Architecture

The LSTM architecture connects the prior cell state (C_{t-1}) to the current cell state (C_t), establishing a pathway that facilitates the seamless transfer of information between consecutive time steps. The cell state functions as a memory, capable of retaining information for extended durations.

LSTM consists of three primary gates: the input gate, the output gate, and the forget gate [36]. Input gate controls the incorporation of new data into the cell state, while the output gate determines which data will be produced as output from the information that is not forgotten and the information from

the input gate in the cell state. Forget Gate controls whether the old information will be ignored or remain relevant for the following process.

BiLSTM addresses a limitation in recurrent neural networks (RNN) and LSTM models. Unlike these models, where information flows only forward, a bidirectional LSTM (BiLSTM) is made up of two LSTMs: one that processes past data sequentially and another that processes future data in reverse order [37]. This bidirectional approach allows for storing text information in both directions and supports further training procedures, leading to improved performance. The outputs of both LSTM networks are integrated for every time sequence.

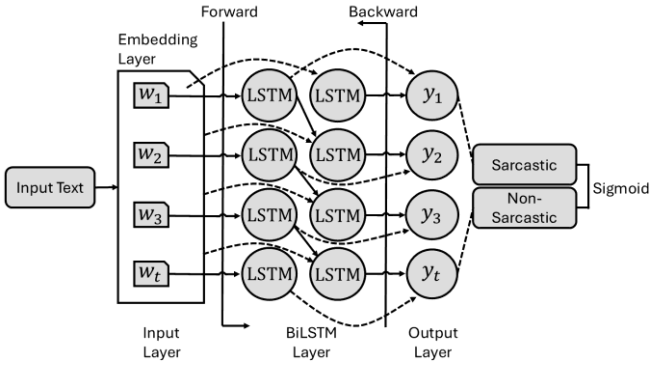


Fig. 6. Deep Learning Layer using BiLSTM

Fig. 6 outlines the structure of BiLSTM, demonstrating that the forward layer adheres to the conventional sequence of a standard LSTM network, whereas the backward layer processes in the opposite direction. The forward layer computes the sequence of $t-1$, t, and $t+1$. In contrast, with the backward layer, output and hidden layer go from $t+1$ to t then to $t-1$. $(\overrightarrow{h_t})$ represent the forward layer and $(\overleftarrow{h_t})$ represent backward layer. As per [38], The mechanisms underlying both the forward and backward LSTM networks can be formally articulated as follows.

$$\overrightarrow{h_t} = LSTM(x_t, h_{t-1}) \quad (1)$$

$$\overleftarrow{h_t} = LSTM(x_t, h_{t+1}) \quad (2)$$

As described in Fig. 6, The hidden layer are connected through bidirectional pathways—moving forward and backward—culminating in the generation of an output. The output value is calculated using the following equation.

$$y_t = U_y\overrightarrow{h_t} + W_y\overleftarrow{h_t} + b_y \quad (3)$$

Let y_t represent the final output value, with U_y and W_y denoting the weight values for the output gate applied to $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$, respectively.

2.5. Evaluation

Assessing the performance of the developed model was essential to ensure its quality. In this study, evaluation was conducted using the Confusion Matrix method, a widely adopted approach to assess how well binary classification models perform [39]. A Confusion Matrix serves as a tabular tool for assessing how a classification model performs in binary classification tasks by comparing predicted outcomes with actual labels [40]. This approach offers important information regarding the model's performance metrics, including precision, recall, accuracy, and the F1-score, helping to determine its overall effectiveness.

TABLE II
CONFUSION MATRIX

		Predicted Label	
		Sarcastic (1)	Non-Sarcastic (0)
True Label	Sarcastic (1)	*TP*	*FN*
	Non-Sarcastic (0)	*FP*	*TN*

Referring to Table II above, the classification outcomes produced by the model are assessed against the actual labels using a confusion matrix, which consists of four components: True Positive (*TP*), False Positive (*FP*), True Negative (*TN*), and False Negative (*FN*). Through this matrix, various performance indicators—including precision, recall, accuracy, and F1-score—can be calculated to provide a thorough evaluation of how well the model performs [41].

Accuracy measured how accurately the model classified the entire dataset. Recall measures how many of the actual positive instances were correctly identified, calculated as the ratio of true positives to the total number of actual positives. Precision, on the other hand, assesses the accuracy of positive predictions by computing the ratio of true positives to the total predicted positives. The F1-Score represents a harmonic mean that balances both precision and recall [42]. The standard equations used to compute accuracy, recall, precision, and F1-Score, all derived from the confusion matrix, are provided below.

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

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$F1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (7)$$

III. RESULTS

3.1. Text Preprocessing

Preprocessing is crucial for allowing textual information to be understood and handled effectively by text analysis systems. Table III below presents the text preprocessing steps undertaken.

TABLE III
TEXT PREPROCESSING

Process	Result
Input	Sumpah dah indihome knp sih? udh 3 hari ini lelet bgt jaringannya 😭
Cleaning	Sumpah dah indihome knp sih udh hari ini lelet bgt jaringannya
Case Folding	sumpah dah indihome knp sih udh hari ini lelet bgt jaringannya
Tokenizing	['sumpah', 'dah', 'indihome', 'knp', 'sih', 'udh', 'hari', 'ini', 'lelet', 'bgt', 'jaringannya']
Spelling Normalization	['sumpah', sudah, 'indihome', kenapa, 'sih', sudah, 'hari', 'ini', lambat, sekali, 'jaringannya']
Stopword Removal	['indihome', 'lambat', 'jaringannya']
Stemming	['indihome', 'lambat', 'jaring']

Following Table III, text preprocessing is done by including several steps, such as data cleaning involves eliminating elements like hashtags, web links, user mentions, and special characters to generate raw content. Subsequently, case folding involves converting every character in the text into lowercase letters, boosting the precision of text categorization and maintaining uniformity across the dataset. Tokenizing, which breaks text into words based on space delimiters, serves to prepare the data for easy computer reading, while spelling normalization involves correcting or replacing words with spelling errors or abbreviations. Stopword removal eliminates words that lack significant contribution to analysis through the NLTK library, whereas stemming removes affixes to yield base words utilizing the Sastrawi library. The steps outlined are designed to streamline the data, enhance its quality, and ready the textual information for subsequent analysis.

3.2. Imbalanced Data

Imbalanced data denotes a data set characterized by an unequal distribution of its classes. The majority is the class with the greatest number, while the class with a lower number is called the minority.

This situation can be problematic as data imbalance can result in the model tending to learn more about the majority data, which can have an impact on the model's efficacy, particularly when applying it to minority classes. The impact of this imbalance is a decrease in model performance in recognizing and classifying minority data.

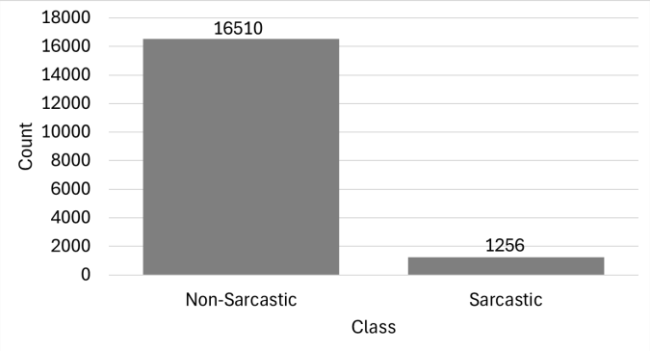


Fig. 7. Class Distribution

In the present investigation, a salient observation emerged from the data distribution depicted in Fig. 7, where the minority class was represented by a mere 1,256 instances, forming approximately 7.07% of the entire dataset. This stark class imbalance underscored the necessity for deliberate and effective strategies to rectify the unbalanced distribution.

Three oversampling techniques were employed systematically to augment minority-class instances. Subsequently, they were combined within the framework of the BiLSTM model.

Subsequently, these oversampling techniques were systematically integrated with a set of fundamental hyperparameters within the framework of the BiLSTM model. These hyperparameters, encompassing a dropout rate of 0.1, an optimizer set to Adam, 32 neurons in the dense layer, 32 neurons in the BiLSTM layer, a batch size of 16, and a learning rate of 0.01, were then randomized and combined during the BiLSTM Testing with Hyperparameter Tuning phase. However, it is essential to clarify that this step was exclusively focused on determining the most effective oversampling technique for the model without engaging in the comprehensive hyperparameter tuning process at this stage. This approach ensured that the impact of each oversampling technique could be assessed independently, providing a clearer understanding of its contribution to model performance. The subsequent phases of experimentation would further refine hyperparameter configurations to optimize the overall effectiveness of the BiLSTM model.

TABLE IV
RESULTS OF BILSTM TESTING WITH OVERSAMPLING TECHNIQUES

Over-sampling Technique	Label	Precision	Recall	F1-Score	Accuracy
SMOTE	Non-Sarcastic	87.4%	82.3%	84.8%	85.2%
	Sarcastic	83.3%	88.2%	85.6%	
ADASYN	Non-Sarcastic	84.6%	83.5%	84.0%	84.0%
	Sarcastic	83.5%	84.6%	84.0%	
ROS	Non-Sarcastic	99.2%	90.1%	94.5%	**94.7%**
	Sarcastic	**90.9%**	**99.3%**	**94.9%**	

The efficacy metrics of BiLSTM testing with various oversampling techniques are delineated in Table IV, focusing on recall, precision, and F1-score for sarcastic and non-sarcastic labels. Among the oversampling methods employed, ROS stood out with notably superior results, particularly in addressing the classification of sarcastic instances. For sarcastic instances, ROS achieved a recall of 99.3%, a precision of 90.9%, and a F1-score of 94.9%. Comparatively, SMOTE and ADASYN exhibited slightly lower performance metrics for the sarcastic class. The overall accuracy score further reinforced the efficacy of ROS, as it attained the highest accuracy (94.7%) among the oversampling techniques evaluated. Consequently, based on these comprehensive evaluation results, the researcher opted for ROS as the preferred oversampling method for its exceptional performance in handling sarcastic instances within the BiLSTM model.

3.3. Results of the BiLSTM Model

After the preprocessing phase, the dataset underwent a meticulous division into three distinct subsets: 80% of the data is designated for training, 10% for validation, and the remaining 10% is for testing. Following this preparatory phase, the BiLSTM was subjected to rigorous evaluation. The crux of this evaluation rested on exploring various hyperparameter combinations within the BiLSTM model, aiming to identify the optimal set that yielded the most favorable performance metrics.

TABLE V
BILSTM PARAMETER

Parameter	Value
BiLSTM Layer Neurons	32, 64, 128
Dense Layer Neurons	32, 64, 128
Dropout Rate	0.1, 0.2, 0.3, 0.4, 0.5
Optimizer	Adam
Learning Rate	0.01, 0.001, 0.0001
Batch Size	16, 32, 64, 128
Epochs	100

Notably, the predetermined values for these hyperparameters, as specified in Table V, played a crucial role in guiding the extensive testing regimen. The training process will undergo the EarlyStopping function, striking a balance between model convergence and the prevention of overtraining. This methodological framework facilitated a comprehensive exploration of the BiLSTM model's hyperparameter landscape, ultimately guiding the selection of the most optimal parameter combination through evaluation.

TABLE VI
BEST 10 RESULTS OF BILSTM TESTING WITH HYPERPARAMETER TUNING

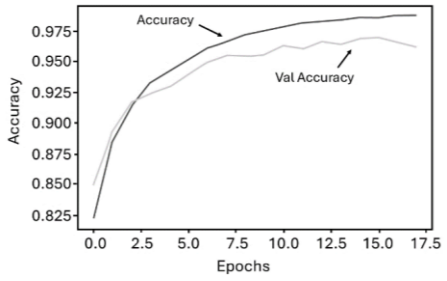
Hyperparameter	Precision	Recall	F1-Score	Accuracy
HP-01	**95.12%**	**99.93%**	**97.47%**	**97.40%**
HP-02	94.80%	99.72%	97.20%	97.13%
HP-03	94.73%	99.52%	97.06%	96.99%
HP-04	94.90%	99.31%	97.06%	96.99%
HP-05	94.90%	99.31%	97.06%	96.99%
HP-06	94.19%	100.00%	97.01%	96.92%
HP-07	94.31%	99.86%	97.01%	96.92%
HP-08	94.31%	99.79%	96.97%	96.88%
HP-09	94.07%	100.00%	96.95%	96.85%
HP-10	95.13%	98.82%	96.94%	96.88%

Table VI presents the top 10 BiLSTM results based on hyperparameter tuning, identified by unique configuration IDs. Table VII provides a comprehensive reference for each configuration, detailing the specific hyperparameter values used in the experiments.

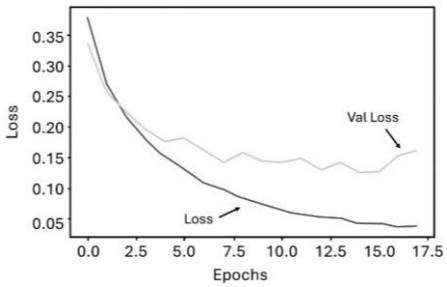
TABLE VII
HYPERPARAMETER REFERENCE FOR TOP 10 BILSTM CONFIGURATIONS

ID	Batch Size	BiLSTM Neurons	Dropout Rate	LR	Dense Neurons	Optimizer
HP-01	16	128	0.3	0.0001	32	Adam
HP-02	32	128	0.1	0.0001	128	Adam
HP-03	16	64	0.2	0.0001	64	Adam
HP-04	16	128	0.1	0.001	32	Adam
HP-05	16	128	0.3	0.0001	64	Adam
HP-06	32	64	0.1	0.0001	64	Adam
HP-07	16	64	0.4	0.001	128	Adam
HP-08	64	128	0.4	0.0001	32	Adam
HP-09	32	128	0.1	0.01	64	Adam
HP-10	16	128	0.2	0.01	32	Adam

As shown in Table VII, the configuration labeled HP-01, which utilizes 128 BiLSTM neurons, a batch size of 16, a dropout rate of 0.3, a learning rate of 0.0001, 32 dense neurons, and the Adam optimizer, was identified as the most optimal setting. This model exhibited an impressive accuracy of 97.40%. The recall, precision, and F1-score for each class further underscore its effectiveness. The F1-score was 97.47%, the recall was 99.93%, and the precision was 95.12% for the sarcastic class. The overall accuracy reinforces the robustness of the selected hyperparameter configuration, showcasing its ability to provide accurate predictions across both classes.



(a) Accuracy Metrics



(b) Loss Metrics

Fig. 8. Metrics of the Optimal BiLSTM Model

Fig. 8 illustrates the graphical representation of the accuracy and loss metrics for the best-performing BiLSTM model. The accuracy graph consistently showed an upward trend, indicating a continuous improvement in the model's ability to make correct predictions. This upward trajectory signified the effectiveness of the selected hyperparameter configuration, showcasing its capacity to improve the model's overall efficacy. Conversely, the loss graph exhibited a steady decline, indicating a consistent decrease in the model's error. The chosen hyperparameters' ability to minimize prediction errors was supported by the decreasing trend in loss. The stability observed in both accuracy and loss graphs suggested that the model, configured with a BiLSTM layer with 128 neurons, a dropout rate of 0.3, a dense layer with 32 neurons, an Adam optimizer, a batch size of 16 with a learning rate of 0.0001 consistently maintained high accuracy while effectively minimizing loss. This reinforced the reliability and efficiency of the identified hyperparameter combination for optimal model performance.

IV. DISCUSSION

This study identified a prominent issue of class imbalance, with the minority class constituting only 7.07% of the dataset (Fig. 7). This substantial disparity might result in biased model predictions, since the model may preferentially favor the dominant class. Resolving this problem is essential for enhancing the model's efficacy, especially in identifying minority-class cases, which are sarcastic comments in this context.

Three methods were systematically integrated into the BiLSTM model, and their effectiveness was evaluated. The results in Table IV reveal that ROS outperformed SMOTE and ADASYN across all performance metrics. ROS achieved higher recall, precision, and F1-score and yielded the best overall accuracy. Precision of 90.9% with ROS indicates a higher rate of correctly predicted sarcastic comments compared to studies where SMOTE and ADASYN were employed. The recall of 99.3% demonstrates ROS's effectiveness in identifying nearly all existing sarcastic comments, a notable improvement over the results typically reported with SMOTE and ADASYN. Furthermore, the F1-score of 94.9% and overall accuracy of 94.7% highlight ROS's robustness in maintaining a balance between recall and precision, thereby ensuring more reliable performance.

The exceptional efficacy of ROS is due to its capacity to produce synthetic samples that more accurately resemble the minority class, hence improving the model's learning and generalization from these examples. This finding aligns with other studies highlighting ROS's effectiveness in handling

class imbalances in various domains [43].

In contrast, SMOTE and ADASYN, while effective in balancing the classes, did not perform as well as ROS. SMOTE achieved an F1-score of 85.6% for the sarcastic class, whereas ADASYN reached 84.0%. This discrepancy may be due to the inherent differences in how these techniques generate synthetic samples. SMOTE generates synthetic examples by interpolating between minority class instances along the line segments connecting them, which can sometimes lead to less diverse samples. ADASYN adjusts the number of synthetic instances in accordance with the distribution density of the minority class samples; however, it may produce samples that are less diverse or representative compared to those generated by ROS. Our findings challenge the prevailing assumption that more sophisticated oversampling techniques like SMOTE and ADASYN are universally superior, suggesting that the selection of an oversampling method must be meticulously evaluated in relation to the characteristics of the dataset and the classification objective involved.

Table VI presents the results of BiLSTM testing with various hyperparameter combinations. The optimal configuration included a batch size of 16, 32 neurons in the dense layer, 128 neurons in the BiLSTM layer, an Adam optimizer, dropout rate of 0.3, and a learning rate of 0.0001, resulting in an accuracy of 97.40%. The metrics of recall, precision, and F1-score for this configuration were notably high, underscoring the significance of careful hyperparameter tuning. The selected hyperparameters effectively balanced the model's complexity and its capacity to learn from the training data, which is consistent with findings from other studies on the importance of hyperparameter tuning in deep learning models [44].

Moreover, while the chosen hyperparameter configuration yielded high performance, it is important to recognize that hyperparameter tuning is a computationally intensive process. Future studies might benefit from automated hyperparameter optimization techniques to streamline this process and potentially discover even better configurations.

The significance of optimizing hyperparameters and resolving class imbalances in machine learning models is underscored by the findings of this study. The superior performance of ROS and the identified optimal hyperparameter configuration underscore the potential for these strategies to enhance model accuracy and robustness. These results have substantial implications for the applicability of BiLSTM models in a variety of fields, particularly those that involve imbalanced datasets, including sentiment analysis, medical diagnosis, and fraud detection.

V. Conclusion

This study provides a comprehensive approach to sarcasm detection in Indonesian social media through the integration of advanced techniques such as BiLSTM, oversampling methods, and hyperparameter tuning. Oversampling was necessary as a result of the dataset's substantial class imbalance. ROS handled sarcasm best; however, the application of ROS in text analysis should be approached cautiously due to the possibility of sample repetition, which may result in identical training and testing datasets, which can compromise the effectiveness of the training processes and undermine the reliability of predictive outcomes.

Furthermore, the study identified an optimal hyperparameter set for the BiLSTM model, which resulted in excellent accuracy, recall, precision, and F1-score for both sarcastic and non-sarcastic instances. The selected hyperparameter configuration showed resilience in predicting both classes, with 97.40% accuracy, 95.12% precision, 99.93% recall, and F1-score of 97.47% for the sarcastic class.

Besides improving our understanding of sarcasm in text, this research opens opportunities for more effective communication analysis in the field of Internet services and beyond. Future directions should focus on exploring innovative techniques to even better detect sarcasm and improve sentiment analysis, especially in different linguistic and cultural contexts. In addition, valuable insights for enhancing the user experience and service quality could be obtained by investigating the practical implementation of these findings in real-world scenarios, such as analyzing consumer feedback at ISPs.

Acknowledgment

The authors express their gratitude to Ivosight Solusi Data for their assistance in obtaining consumer feedback regarding Indonesia's most prominent internet service providers via Twitter. This assistance has been highly valuable in facilitating the completion of this research and contributing to the author's academic work.

References

[1] DataIndonesia.id, “APJII: Pengguna Internet Indonesia 215,63 Juta pada 2022-2023,” https://dataindonesia.id/digital/detail/apjii-pengguna-internet-indonesia-21563-juta-pada-20222023.

[2] Detikinet, “Kecepatan Internet Indonesia Peringkat Terakhir di Asia Tenggara,” https://inet.detik.com/telecommunication/d-6639503/kecepatan-internet-indonesia-peringkat-terakhir-di-asia-tenggara.

[3] S. K. Bharti, R. K. Gupta, P. K. Shukla, W. A. Hatamleh, H. Tarazi, and S. J. Nuagah, “Multimodal Sarcasm Detection: A Deep Learning Approach,” *Wirel Commun Mob Comput*, vol. 2022, pp. 1–10, May 2022, doi: 10.1155/2022/1653696.

[4] N. Ganguly and P. Kumaraguru, “The positive and negative effects of social media in India,” *Commun ACM*, vol. 62, no. 11, pp. 98–99, Oct. 2019, doi: 10.1145/3345671.

[5] A. Onan and M. A. Tocoglu, “A Term Weighted Neural Language Model and Stacked Bidirectional LSTM Based Framework for Sarcasm Identification,” *IEEE Access*, vol. 9, pp. 7701–7722, 2021, doi: 10.1109/ACCESS.2021.3049734.

[6] A. Khatri and P. P, “Sarcasm Detection in Tweets with BERT and GloVe Embeddings,” in *Proceedings of the Second Workshop on Figurative Language Processing*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2020, pp. 56–60. doi: 10.18653/v1/2020.figlang-1.7.

[7] A. Kumar, V. T. Narapareddy, V. Aditya Srikanth, A. Malapati, and L. B. M. Neti, “Sarcasm Detection Using Multi-Head Attention Based Bidirectional LSTM,”

IEEE Access, vol. 8, pp. 6388–6397, 2020, doi: 10.1109/ACCESS.2019.2963630.

[8] S. M. Sarsam, H. Al-Samarraie, A. I. Alzahrani, and B. Wright, "Sarcasm detection using machine learning algorithms in Twitter: A systematic review," *International Journal of Market Research*, vol. 62, no. 5, pp. 578–598, Sep. 2020, doi: 10.1177/1470785320921779.

[9] J. Godara, R. Aron, and M. Shabaz, "Sentiment analysis and sarcasm detection from social network to train health-care professionals," *World Journal of Engineering*, vol. 19, no. 1, pp. 124–133, Feb. 2022, doi: 10.1108/WJE-02-2021-0108.

[10] M. Bhakuni, K. Kumar, Sonia, C. Iwendi, and A. Singh, "Evolution and Evaluation: Sarcasm Analysis for Twitter Data Using Sentiment Analysis," *J Sens*, vol. 2022, pp. 1–10, Oct. 2022, doi: 10.1155/2022/6287559.

[11] C. I. Eke, A. A. Norman, and L. Shuib, "Multi-feature fusion framework for sarcasm identification on twitter data: A machine learning based approach," *PLoS One*, vol. 16, no. 6, p. e0252918, Jun. 2021, doi: 10.1371/journal.pone.0252918.

[12] A. Kumar and G. Garg, "Empirical study of shallow and deep learning models for sarcasm detection using context in benchmark datasets," *J Ambient Intell Humaniz Comput*, vol. 14, no. 5, pp. 5327–5342, May 2023, doi: 10.1007/s12652-019-01419-7.

[13] R. Haque, S. H. Laskar, K. G. Khushbu, M. J. Hasan, and J. Uddin, "Data-Driven Solution to Identify Sentiments from Online Drug Reviews," *Computers*, vol. 12, no. 4, p. 87, Apr. 2023, doi: 10.3390/computers12040087.

[14] Y. Bin, Y. Yang, F. Shen, N. Xie, H. T. Shen, and X. Li, "Describing Video With Attention-Based Bidirectional LSTM," *IEEE Trans Cybern*, vol. 49, no. 7, pp. 2631–2641, Jul. 2019, doi: 10.1109/TCYB.2018.2831447.

[15] Y. Cai, H. Cai, and X. Wan, "Multi-Modal Sarcasm Detection in Twitter with Hierarchical Fusion Model," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2019, pp. 2506–2515. doi: 10.18653/v1/P19-1239.

[16] G. Liu and J. Guo, "Bidirectional LSTM with attention mechanism and convolutional layer for text classification," *Neurocomputing*, vol. 337, pp. 325–338, Apr. 2019, doi: 10.1016/j.neucom.2019.01.078.

[17] A. Alqahtani *et al.*, "An efficient approach for textual data classification using deep learning," *Front Comput Neurosci*, vol. 16, Sep. 2022, doi: 10.3389/fncom.2022.992296.

[18] M. D. Hilmawan, "Deteksi Sarkasme Pada Judul Berita Berbahasa Inggris Menggunakan Algoritme Bidirectional LSTM," *Journal of Dinda : Data Science, Information Technology, and Data Analytics*, vol. 2, no. 1, pp. 46–51, Feb. 2022, doi: 10.20895/dinda.v2i1.331.

[19] L. Hermawan and M. B. Ismiati, "Pembelajaran Text Preprocessing berbasis Simulator Untuk Mata Kuliah Information Retrieval," *Jurnal Transformatika*, vol. 17, no. 2, p. 188, Jan. 2020, doi: 10.26623/transformatika.v17i2.1705.

[20] C. P. Chai, "Comparison of text preprocessing methods," *Nat Lang Eng*, vol. 29, no. 3, pp. 509–553, May 2023, doi: 10.1017/S1351324922000213.

[21] D. Normawati and S. A. Prayogi, "Implementasi Naïve Bayes Classifier dan Confusion Matrix pada analisis sentimen berbasis teks pada Twitter," *J-SAKTI (Jurnal Sains Komputer Dan Informatika)*, vol. 5, no. 2, pp. 697–711, 2021, doi: https://doi.org/10.30645/j-sakti.v5i2.369.

[22] Y. HaCohen-Kerner, D. Miller, and Y. Yigal, "The influence of preprocessing on text classification using a bag-of-words representation," *PLoS One*, vol. 15, no. 5, p. e0232525, May 2020, doi: 10.1371/journal.pone.0232525.

[23] J. Yan, "Text Mining with R: A Tidy Approach, by Julia Silge and David Robinson. Sebastopol, CA: O'Reilly Media, 2017. ISBN 978-1-491-98165-8. XI + 184 pages.," *Nat Lang Eng*, vol. 28, no. 1, pp. 137–139, Jan. 2022, doi: 10.1017/S1351324920000649.

[24] C. Wan, Y. Wang, Y. Liu, J. Ji, and G. Feng, "Composite Feature Extraction and Selection for Text Classification," *IEEE Access*, vol. 7, pp. 35208–35219, 2019, doi: 10.1109/ACCESS.2019.2904602.

[25] S. Memon, G. Ali, K. N. M. -, A. Shaikh, S. K.Aasoori, and F. Ul, "Comparative Study of Truncating and Statistical Stemming Algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 2, 2020, doi: 10.14569/IJACSA.2020.0110272.

[26] L. Ronghui and W. Xinhong, "Application of Improved Convolutional Neural Network in Text Classification," *IAENG Int J Comput Sci*, vol. 49, no. 3, pp. 762–767, 2022.

[27] E. Chersoni, E. Santus, C.-R. Huang, and A. Lenci, "Decoding Word Embeddings with Brain-Based Semantic Features," *Computational Linguistics*, vol. 47, no. 3, pp. 663–698, Nov. 2021, doi: 10.1162/coli_a_00412.

[28] C. Fu and J. Yang, "Granular Classification for Imbalanced Datasets: A Minkowski Distance-Based Method," *Algorithms*, vol. 14, no. 2, p. 54, Feb. 2021, doi: 10.3390/a14020054.

[29] T. Hasanin, T. M. Khoshgoftaar, J. L. Leevy, and R. A. Bauder, "Investigating class rarity in big data," *J Big Data*, vol. 7, no. 1, p. 23, Dec. 2020, doi: 10.1186/s40537-020-00301-0.

[30] K. R. M. Fernando and C. P. Tsokos, "Dynamically Weighted Balanced Loss: Class Imbalanced Learning and Confidence Calibration of Deep Neural Networks," *IEEE Trans Neural Netw Learn Syst*, vol. 33, no. 7, pp. 2940–2951, Jul. 2022, doi: 10.1109/TNNLS.2020.3047335.

[31] Google for Developers, "Imbalanced Data," https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data.

[32] Z. A. Khan, M. Adil, N. Javaid, M. N. Saqib, M. Shafiq, and J.-G. Choi, "Electricity Theft Detection Using Supervised Learning Techniques on Smart Meter Data," *Sustainability*, vol. 12, no. 19, p. 8023, Sep. 2020, doi: 10.3390/su12198023.

[33] T. Wongvorachan, S. He, and O. Bulut, "A Comparison of Undersampling, Oversampling, and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining,"

Information, vol. 14, no. 1, p. 54, Jan. 2023, doi: 10.3390/info14010054.

[34] R. Soundrapandiyan, A. Manickam, M. Akhloufi, Y. V. S. Murthy, R. D. M. Sundaram, and S. Thirugnanasambandam, "An Efficient COVID-19 Mortality Risk Prediction Model Using Deep Synthetic Minority Oversampling Technique and Convolution Neural Networks," *BioMedInformatics*, vol. 3, no. 2, pp. 339–368, May 2023, doi: 10.3390/biomedinformatics3020023.

[35] Haibo He, Yang Bai, E. A. Garcia, and Shutao Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, IEEE, Jun. 2008, pp. 1322–1328. doi: 10.1109/IJCNN.2008.4633969.

[36] J. Wu, J. Yuan, Y. Weng, and R. Ayyanar, "Spatial-Temporal Deep Learning for Hosting Capacity Analysis in Distribution Grids," *IEEE Trans Smart Grid*, vol. 14, no. 1, pp. 354–364, Jan. 2023, doi: 10.1109/TSG.2022.3196943.

[37] Q. Wang, Y. Yu, H. O. A. Ahmed, M. Darwish, and A. K. Nandi, "Open-Circuit Fault Detection and Classification of Modular Multilevel Converters in High Voltage Direct Current Systems (MMC-HVDC) with Long Short-Term Memory (LSTM) Method," *Sensors*, vol. 21, no. 12, p. 4159, Jun. 2021, doi: 10.3390/s21124159.

[38] Z. Hameed and B. Garcia-Zapirain, "Sentiment Classification Using a Single-Layered BiLSTM Model," *IEEE Access*, vol. 8, pp. 73992–74001, 2020, doi: 10.1109/ACCESS.2020.2988550.

[39] M. Fahmy Amin, "Confusion Matrix in Binary Classification Problems: A Step-by-Step Tutorial," *Journal of Engineering Research*, vol. 6, no. 5, pp. 0–0, Dec. 2022, doi: 10.21608/erjeng.2022.274526.

[40] M. Fahmy Amin, "Confusion Matrix in Binary Classification Problems: A Step-by-Step Tutorial," *Journal of Engineering Research*, vol. 6, no. 5, pp. 0–0, Dec. 2022, doi: 10.21608/erjeng.2022.274526.

[41] J. Erbani, P.-É. Portier, E. Egyed-Zsigmond, and D. Nurbakova, "Confusion Matrices: A Unified Theory," *IEEE Access*, vol. 12, pp. 181372–181419, 2024, doi: 10.1109/ACCESS.2024.3507199.

[42] L. Lavazza and S. Morasca, "Comparing ϕ and the F-measure as performance metrics for software-related classifications," *Empir Softw Eng*, vol. 27, no. 7, p. 185, Dec. 2022, doi: 10.1007/s10664-022-10199-2.

[43] H. Rathpisey and T. B. Adji, "Handling Imbalance Issue in Hate Speech Classification using Sampling-based Methods," in *2019 5th International Conference on Science in Information Technology (ICSITech)*, IEEE, Oct. 2019, pp. 193–198. doi: 10.1109/ICSITech46713.2019.8987500.

[44] L. Liao, H. Li, W. Shang, and L. Ma, "An Empirical Study of the Impact of Hyperparameter Tuning and Model Optimization on the Performance Properties of Deep Neural Networks," *ACM Transactions on Software Engineering and Methodology*, vol. 31, no. 3, pp. 1–40, Jul. 2022, doi: 10.1145/3506695.