Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

Hate Speech Classification in TikTok Reviews Using TF-IDF, Word2Vec, and Differential Evolution with RNN

Rizkialdy Fatha*, Yuliant Sibaroni, Sri Suryani Prasetyowati

School of Computing, Informatics, Telkom University, Bandung, Indonesia
Email: rizuki@student.telkomuniversity.ac.id, yuliant@telkomuniversity.ac.id, srisuryani@telkomuniversity.ac.id
Correspondence Author Email: rizuki@student.telkomuniversity.ac.id
Submitted: **08/07/2024**; Accepted: **08/09/2024**; Published**: 09/09/2024**

Abstract-In In the ever-evolving digital era, social media, especially platforms like TikTok, have become primary channels for users to share opinions, experiences, and expressions. However, the increasing prevalence of hate speech in reviews on the Google Play Store for the TikTok app indicates the need for a sophisticated approach to identify and classify harmful content. This research aims to optimize the classification of hate speech in Google Play reviews of the TikTok app by integrating Term Frequency-Inverse Document Frequency (TF-IDF), Differential Evolution, and Word2Vec within a Recurrent Neural Network (RNN) model. The TF-IDF technique is used to extract relevant features from reviews, while Differential Evolution efficiently optimizes the model parameters. Word2Vec enhances the representation of words in the context of app reviews, and the RNN model enables the recognition of temporal patterns in hate speech. The results of this research, achieving the highest accuracy of 88.63% and an F1 score of 88.62%, are expected to contribute significantly to improving hate speech classification on digital platforms focused on app reviews. The study demonstrates the effectiveness of combining advanced feature extraction and optimization techniques to develop a robust classification system for identifying and mitigating hate speech.

Keywords: Classification; Term Frequency-Inverse Document Frequency; Word2Vec; Differential Evolution; Recurrent Neural Network

1. INTRODUCTION

On the era of online interactions, social media platforms like TikTok have become central entertainment hubs where users can share their opinions and experiences. These platforms have revolutionized the way people connect, interact, and entertain themselves. With the convenience and widespread reach of these platforms, millions of users, including teenagers, are engaged daily in creating and consuming content. This massive user engagement has brought about a significant shift in digital communication and content-sharing practices. However, along with this rapid growth and popularity, a negative phenomenon related to hate speech has emerged, particularly in reviews discussing TikTok. The increase in user activity, especially among teenagers, has unfortunately led to a rise in negative interactions. Hate speech, characterized by derogatory, offensive, and inflammatory language, not only disrupts user experiences but also creates an unsafe and negative online environment [1]. This problem is not just a minor inconvenience but a serious issue that affects the mental well-being of users and the overall health of online communities.In light of research on hate speech in TikTok app reviews, the development of advanced classification methods becomes increasingly important [1].

The importance of addressing hate speech cannot be overstated. Hate speech has the potential to escalate into real-world consequences, including bullying, harassment, and even physical violence. Therefore, in light of research on hate speech in TikTok app reviews, the development of advanced classification methods becomes increasingly important [1]. Accurate and efficient identification of hate speech is crucial for maintaining a positive user experience and ensuring the safety of users. This challenge has spurred researchers and technologists to explore innovative approaches to detect and mitigate hate speech effectively [2]. One of the primary tools in this endeavor is the development of sophisticated machine learning algorithms capable of analyzing and classifying vast amounts of text data with high accuracy.

In research [3] conducted by Ravinder Ahuja et al., sentiment analysis was performed on the SS-Tweet dataset using two key features, namely TF-IDF and N-Gram. Sentiment analysis involves determining the sentiment expressed in a piece of text, which can range from positive and neutral to negative. This type of analysis is particularly useful in understanding public opinion and identifying problematic content. In this study, six classification algorithms were employed: Decision Tree, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), Logistic Regression (LR), and Naive Bayes (NB). The results indicated that using TF-IDF features at the word level performed 3-4% better than N-Gram features. This finding underscores the effectiveness of TF-IDF in capturing the relevance and importance of words within a document. Based on these results, the study recommends using TF-IDF feature extraction at the word level as the best choice for sentiment analysis using machine learning algorithms [3]. TF-IDF, which stands for Term Frequency-Inverse Document Frequency, is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents. By weighing the frequency of words inversely with their occurrence in the overall corpus, TF-IDF helps in emphasizing words that are significant in a specific context.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

Another noteworthy approach to enhancing classification performance is highlighted in research conducted by K. Vijayaprabakaran et al. This study explored the use of the Differential Evolution (DE) approach to find the optimal activation function in the Long Short-Term Memory Network (LSTM). LSTM is a type of Recurrent Neural Network (RNN) that is particularly effective in handling sequential data and capturing long-term dependencies. The researchers conducted experiments on the IMDB dataset for sentiment classification and the UCI HAR dataset for human activity recognition. The study revealed that the combination of comb-H-sine activation functions yielded better results than other activation functions in terms of accuracy. The statistical test results indicated that comb-H-sine is significantly different from several other activation functions. The conclusion drawn from this study is that searching for the optimal activation function can be effectively achieved using the Differential Evolution approach. This optimization method, which mimics the process of natural selection, is considered to enhance the classification system's performance significantly [4].

By an efforts to expand classification capabilities, the implementation of Word2Vec for feature expansion in RNN (Recurrent Neural Network) represents a significant step [5]. This method utilizes the occurrence of paired words in occurrence matrices to train text, making it efficient in utilizing statistics while also maintaining the linear structure of the Word2vec method [6]. Word2Vec has proven to perform well in tasks like word analogy [6]. It provides the ability to embed the semantic context of words.

Based on research [7] conducted by Yequan Wang et al., this research proposes a capsule model based on Recurrent Neural Networks (RNN-Capsule) for sentiment classification. The model was evaluated using two benchmark datasets, namely Movie Review (MR) and Stanford Sentiment Treebank (SST), as well as one proprietary dataset. Experimental results showed that the RNN-Capsule model achieved the highest accuracy with an accuracy value of 91.6% compared to several other comparative methods on both datasets. This research successfully demonstrated that RNN can provide high accuracy results for predictive modeling. While the RNN approach allows the model to understand the temporal context in text [5].

Combining TF-IDF feature extraction technology, Differential Evolution optimization, and the application of Word2Vec for RNN feature expansion in the classification system this research is aiming to create an effective approach in identifying and classifying hate speech in TikTok app reviews. A deep understanding of this context is expected to contribute significantly toward efforts to create a more positive and safe online environment for the TikTok community.

2. RESEARCH METHODOLOGY

2.1 System Design

Figure 1 presents a comprehensive flowchart that delineates the entire system architecture designed for the classification of hate speech. This visual representation offers a step-by-step breakdown of the processes involved, starting from data collection to the final model evaluation. The flowchart provides an in-depth understanding of how the system efficiently handles and processes data, ensuring accurate classification outcomes.

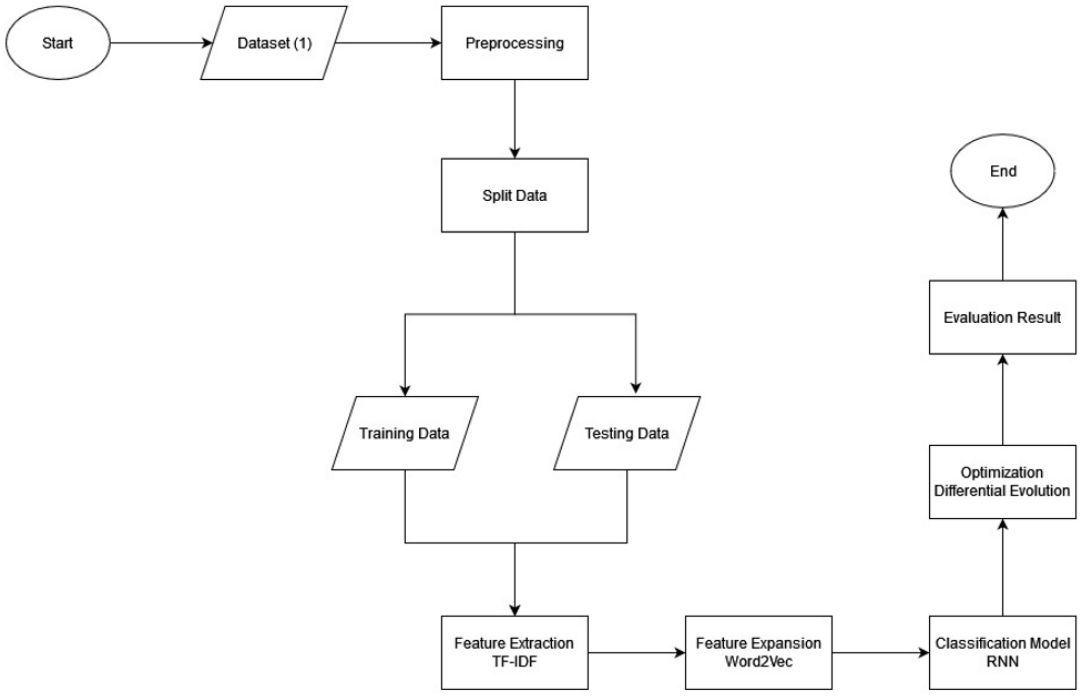

Figure 1. Hate Speech Classification System

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

In Figure 1 it shows the flowchart of the system that's designed for classifying hate speech. In order to get dataset the system must first crawl the data from TikTok app reviews from Google Play Store, after that any duplicate data that have been received will be erased, and finally the cleaned data will be labelled with 0 or 1 where 0 means it's a hate speech and 1 for positive. The labelled data will then undergo to the preprocessing, inside they will undergo normalization, tokenization, data cleaning, and case folding. The preprocessed data will then move on to the feature selection phase for TF-IDF. The data must then be split into two categories: training data and test data. Test data is used to evaluate the performance of the model, while training data is used to train the model. The next step in the modeling process involves using a Recurrent Neural Network (RNN) enhanced by Word2Vec for feature expansion and optimized with Differential Evolution. Subsequently, each model iteration's performance will be assessed by aggregating all of the confusion matrix values. The model's total performance over the whole dataset will be shown in the matrix

2.2 Datasets

In order to do this research. The data crawling process is done using the google play scrapper API in the python programming language. This data crawling produces a dataset in Comma Separated Value (CSV). The datasets were taken from Google Play Store website at the following URL: https://play.google.com/store/apps/details?id=com.ss.android.ugc.trill&hl=en&gl=US&pli=1, under the application name Tiktok, the information that was gathered derived from reviews with scores ranging from 1 to 5. The total data that have been gathered were 15,445 reviews. Following data collecting, a data cleaning procedure is carried out, yielding a total of 15,050 reviews.

The labeling process involves 2 separate individuals working on the datasets. Reviews were labeled based on the sentiment expressed in the text. In Table 1 reviews that contained hate speech, offensive language, or derogatory comments were labeled as 0 (negative), while reviews that were neutral or positive in sentiment were labeled as 1 (positive). The labeling was done manually, ensuring that the context and meaning of the reviews were accurately captured. Both labelers were proficient in the language of the reviews (Indonesian) and worked independently to ensure unbiased labeling. Discrepancies between the labelers' assessments were resolved through discussion and consensus.

Table 1. Labelled Dataset

No.	Review	Label
1	jelek banget anj gatau kenapa jaringan bagus malah lemot ngelag tiba kembali sendiri pas buka tiktok doang yg kek gitu murahan najiss	0 (Negative)
2	aplikasi menghibur dan bisa bantu cari cuan semangat para tik toker semoga dipermudahkan segala urusan dunia akhirat nya dan dilancarkan rezekinya aamin	1 (Positive)
3	apaan makin lama makin jelek bejir masa akun w jadi sepi ada crash kotak masuk padahal udah pencet notifikasi berkali kali	0 (Negative)
4	apk yg bagus tergantung kita sendiri mau lihat video yg bagaimana nya maksudnya yg baik buat kitanya sungguh sangat bermamfaat bagi kita pribadi bisa lihat lantunan bacaan ayat suci al qoran dan sholawat serta pencerahan dari ulama ustadz dan para habaib aamiin	1 (Positive)
5	sumpah ya tiktok di update makin lama makin jelek makin banyak bug nya apa lagi kadang suka berubah rubah update ny pls lah tolong di perbaiki lagi	0 (Negative)

2.3. Data Preprocessing

The preprocessing of data is performed to ensure that the raw data can be utilized effectively for building a more accurate model. This cleaned data is then suitable for further stages. The preprocessing process includes the following steps

a. Data Cleaning this process involves correcting or removing corrupt or inaccurate data to ensure the data is clean and reliable [8]. For data cleaning, we used the Python libraries pandas and numpy. Duplicate reviews were removed, and special characters and excessive whitespace were cleaned.
b. Case Folding this process changes every letter into lowercase. For instance, the phrase "This Is a Test" becomes "this is a test" [9]. This step was implemented using the lower() function in Python.
c. Removing Stopwords this process involves eliminating common words (stopwords) that do not contribute significant information to the analysis. We used the NLTK (Natural Language Toolkit) library's predefined stopwords list for the Indonesian language to perform this task [10]. This process reduces words to their base or root form, helping to decrease the dimensionality of the text data.
d. Stemming this process reduces words to their root form, ensuring that different forms of a word are analyzed as a single item. We used the Sastrawi library, a popular stemming library for the Indonesian language, to perform this step.

Tokenization in this process, text is broken down into individual tokens or separate words, which allows for more detailed processing [11]. We used the nltk library for tokenization.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

2.4. Feature Extraction TF-IDF

Term Frequency-Inverse Document Frequency (TF-IDF) is a well-known method used to evaluate the importance of a word within a document [3]. The term frequency (TF) of a specific word (t) is calculated as the number of times that word appears in the document divided by the total number of words in that document. The Inverse Document Frequency (IDF) measures a word's importance [3]. Some words like "is," "an," "and," etc., appear frequently but are not significant. IDF is calculated using the formula:

$$\mathrm{IDF(t)} = \log\left(\frac{N}{DF}\right) \quad (1)$$

Where N = Total number of documents and DF = Number of documents that contain the term t. TF-IDF combines two different concepts: Term Frequency and Inverse Document Frequency. TF measures how often a word appears in a document, while IDF gives lower weights to words that occur more frequently and higher weights to those that occur rarely. At this stage, the TF-IDF feature is performed on each word appearing in the comments during the weighting stage [12].

There are two main approaches described. The first approach explains the methods of calculating TF and IDF and the fundamental concept of TF-IDF as a way to evaluate the significance of words within a document [3]. The second approach highlights the practical application of TF-IDF in measuring the relevance of keywords across a collection of documents. Both sources provide a balanced view between the theoretical concept and practical implementation of TF-IDF in text analysis and document clustering [12].

2.5. Feature Expansion Word2Vec

Word2Vec is a popular model in the field of Natural Language Processing (NLP) for representing words as vectors. Developed by Mikolov and colleagues, Word2Vec employs a neural network-based approach to produce continuous vector representations of words [13]

Word2Vec features two main architectures: Continuous Bag of Words (CBOW) and Skip-Gram. In the CBOW model, the system predicts a target word based on its context words, whereas in the Skip-Gram model, the system predicts the context words based on a target word [14]. This makes the Skip-Gram model particularly effective in handling less frequent words, as it focuses on predicting the surrounding context for a given word. This architectures is visually summarized in Figure 2, which illustrate the CBOW model and also the Skip-Gram models

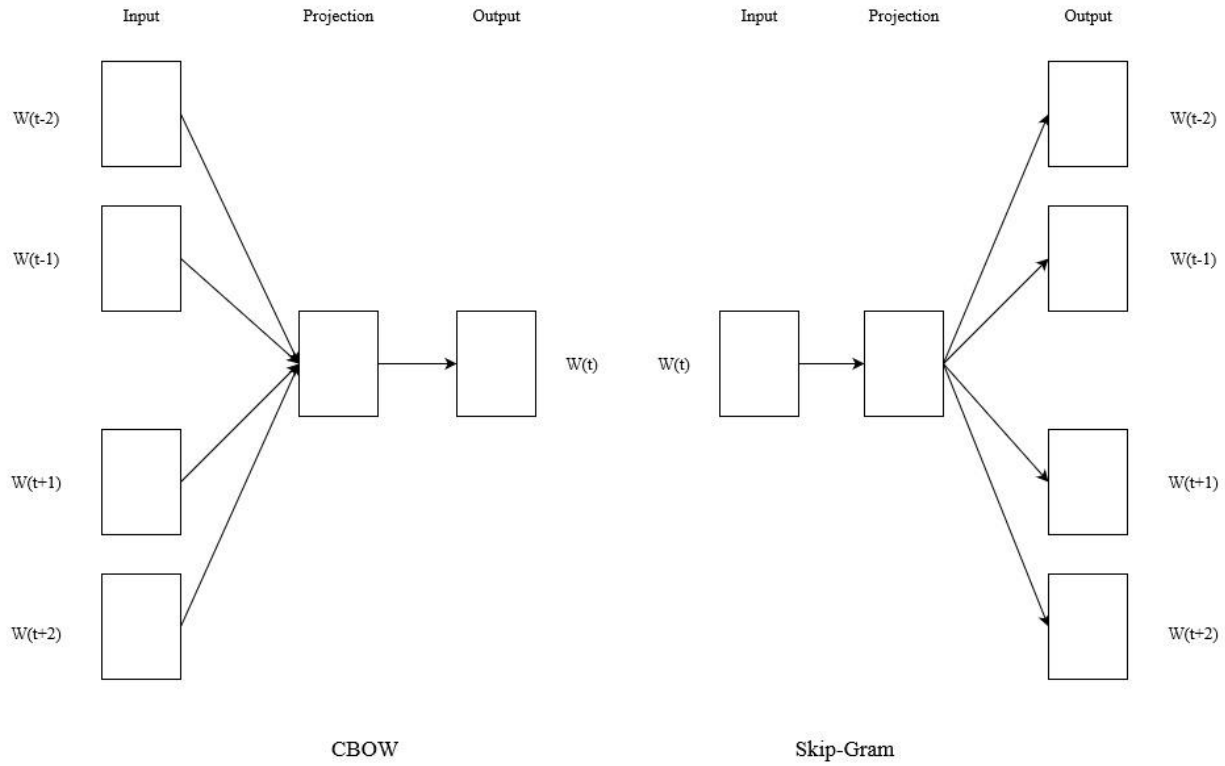

Figure 2. Word2Vec CBOW and Skip-Gram models

Each word in the vocabulary is mapped to a dense vector of real numbers using a shallow probabilistic neural language model [15]. By employing Word2Vec, similar words are positioned closely within the embedding space, meaning that words that share common contexts in the text data are likely to have similar vector representations. This property allows for capturing semantic and syntactic word relationships effectively, making Word2Vec a powerful tool for a variety of NLP applications [15].

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

2.6. Classification Model RNN

Recurrent Neural Network (RNN) is a type of deep learning approach used for sentiment analysis [5]. It generates output based on previous computations using sequential information [5]. Before the advent of RNNs, traditional neural networks used independent inputs that were not suitable for several tasks in Natural Language Processing [5]. For example, predicting words in a given sentence. RNNs are an efficient model for sentiment analysis [5]. They utilize memory cells capable of capturing information about long sequences [5]. This process is illustrated in **Figure 3**

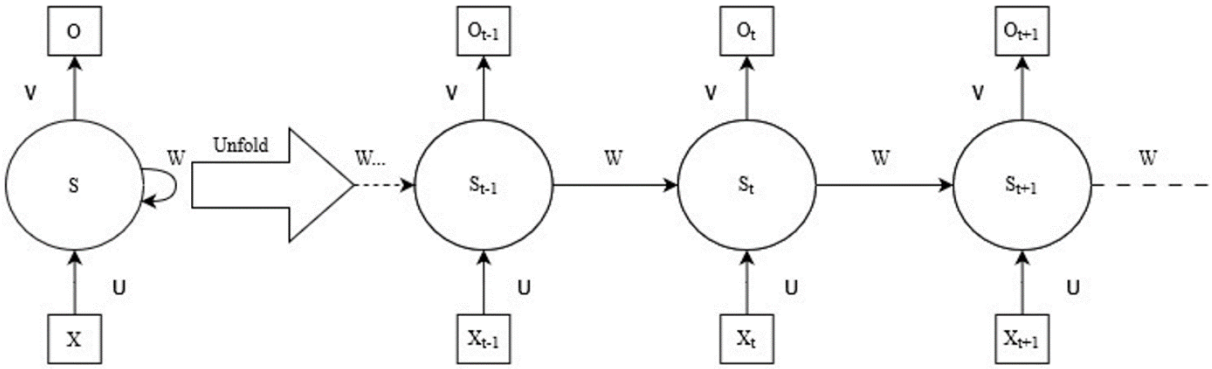

Figure 3. RNN Model

Based on the illustration above we can use the basic formula of RNN:

$$at = f(ht-1, xt) \quad (2)$$

Where function f is typically a tanh function, and xt denotes the sequence of inputs $(x0, x1, x2, \ldots, xt)$ [5]. This formulation captures the essence of how RNNs process data sequentially, integrating past information (through h_{t-1}, the previous hidden state) with current input (xt) to produce the current output at.

An RNN (Recurrent Neural Network) is a type of recurrent neural network that takes sequential data as input, recursively processing in the direction of sequence evolution, with all cyclic elements connected by a chain [16]. RNNs can be bidirectional, allowing them to perform loops [7]. This structure enables RNNs to display dynamic temporal behavior because the connections between units form a directed cycle. Unlike forward neural networks, RNNs can utilize their internal memory to process arbitrary input sequences [16].

It is noted that while RNN models can handle short-term dependencies within sequence data effectively, they face challenges with long-term dependencies due to the problem of gradient explosion [7]. This issue occurs when gradients grow exponentially during backpropagation, leading to unstable training processes and difficulty in learning from data where important information is separated by large gaps.

The number of neurons in each layer is chosen to balance computational efficiency and model performance, avoiding overfitting while ensuring the model's capacity to learn. The tanh activation function helps maintain gradient stability by keeping outputs within a specific range. The learning rate, optimized through Differential Evolution, balances convergence speed and accuracy, while the batch size ensures efficient training and generalization. Dropout regularization is applied to prevent overfitting by randomly setting a fraction of input units to zero during training. The embedding layer parameters include a vocabulary limited to the most frequent words, dense vector representation for each word, and uniform sequence lengths through padding or truncation. Training parameters include a validation split of the training data, a set number of epochs for training, a test size to evaluate the model, and a fixed random state to ensure reproducibility of the train-test split. These parameters were selected based on empirical testing and established practices in the field to ensure the model's effectiveness in hate speech classification.

2.7. Optimization Differential Evolution

Differential Evolution is one of the most popular evolutionary algorithms introduced by Storn and Price [17]. Differential Evolution (DE) is a stochastic, population-based global optimization technique, particularly effective in searching for the global optimum from a large number of candidates (population). The idea of population evolution provides better solutions. DE involves maintaining a population of candidate solutions that undergo mutation, crossover, evaluation, and selection iterations. The evolution starts with the initialization of the population by randomly selecting candidates [4].

The Differential Evolution (DE) algorithm is adopted because it is known for its outstanding performance using various mutation strategies in certain literatures, and it has few parameters to adjust [17]. The mutation process involves three candidates and helps increase the diversity of the population [4]. The crossover approach involves creating new candidate components based on the weighted difference between two randomly selected population members plus a third population member. In this study, the DE algorithm optimizes the learning rate, number of neurons in the RNN layers, and dropout rate. The learning rate determines the step size during the gradient descent

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

optimization process, balancing the speed of convergence and the accuracy of the model. The number of neurons affects the model's capacity to learn patterns from the data, and optimizing this parameter ensures the model captures complex patterns without overfitting. The dropout rate specifies the fraction of neurons to drop during training, and an optimal rate improves the model's generalization ability. These parameters were selected for optimization because they significantly impact the performance and generalization ability of the RNN model, leading to a more accurate and robust hate speech classification model. This disrupts the population members relative to the broader population spread [4].

DE starts by initializing training parameters such as population size N, individual dimension N_par, mutation scaling parameter F, and crossover probability CR [17]. Initially, a population X with size N and dimension N_par is generated using the formula:

$$x_i = L_i + rand(N + N_{par}) * (U_i + L_i), x_i \epsilon X, i = 1,2,\ldots,N, \quad (3)$$

where L and U represent the lower and upper bounds of the search space, respectively. rand(., .) is a function used to generate a random matrix in the interval (0,1) [17].

3. RESULTS AND DISCUSSION

3.1 TikTok Reviews

In this research, the dataset related to the TikTok application was used. Data obtained from Google Play Store Reviews amounted to 15,050 review data. Separated into train and test datasets using an 80:20 distribution ratio, there are up to 12,040 training datasets and 3,010 test datasets overall.

3.2 RNN Modelling

The performance results of the RNN model were evaluated in four scenarios. Each scenario was designed to isolate the effect of different components on the model's performance, aiming to identify the contribution of each component to the overall classification accuracy:

First Scenario This scenario evaluates the model without any feature selection. The purpose is to understand the baseline performance of the RNN using raw data features without any enhancement. The model's performance is assessed using 10-fold cross-validation to provide a reliable evaluation and reduce the risk of overfitting.

Second Scenario In this scenario, TF-IDF feature extraction is applied. The goal is to measure the impact of TF-IDF in improving the model's performance by converting the text data into a format that better represents the importance of words. Cross-validation is utilized to validate the consistency and reliability of the performance metrics.

Third Scenario This scenario utilizes Word2Vec for feature expansion. The aim is to evaluate how well the model performs when word embeddings are used to capture the semantic meaning of the text, providing richer contextual information.

Fourth Scenario Here, Differential Evolution optimization is applied to the model. The objective is to assess the improvements in the model's performance when the hyperparameters (such as learning rate, number of neurons, and dropout rate) are optimized for better accuracy and generalization.

Each scenario's results are compared to determine which combination of features and optimizations yields the best performance for hate speech classification in TikTok reviews.

3.2.1 First Scenario

In this scenario, the model will determine the optimal data diversion by comparing training and test data to achieve the best accuracy that the model can do. The dataset will be split with the ratios of 90:10, and 80:20 initially without any cross folding, then the dataset is evaluated using 10-fold cross-validation, where the data is split into 10 subsets, and the model is trained and tested 10 times, each time using a different subset as the test set and the remaining subsets as the training set. This method ensures that the model's performance is evaluated more reliably and reduces the risk of overfitting. The accuracy and other parameters are presented on Table 2 and Table 3.

Table 2. Result of Scenario 1

Test Size.	Accuracy	F1-Score	Precision	Recall
80:20	84.29%	84.80%	86.17%	84.29%
90:10	84.45%	84.56%	84.71%	84.45%

Table 3. Result of Scenario 1 using Cross Validation

Accuracy	F1-Score	Precision	Recall
86.62%	86.51%	86.48%	86.59%

The results from Table 2 displays the performance metrics of a classification model evaluated using two different test size ratios: 80:20 and 90:10. The results indicate that the RNN model, even without additional feature

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

selection or optimization techniques, can achieve reasonably high accuracy and F1-score. This serves as a baseline performance metric to compare the effects of various feature extraction and optimization methods in subsequent scenarios. The decision to utilize the 10-fold cross-validation was validated by the significant improvements in F1 score and accuracy, which are critical for evaluating the effectiveness of the classification system [18].

3.2.2 Second Scenario

In the second scenario, the utilization of TF-IDF feature extraction using the adjustable max feature was explored. By using this exploration, the model aimed to investigate what will happen when the max feature is adjusted. The results of this scenario, initially presented without cross-validation, are shown in **Table 4** with adjusted features ranging from 500-15000.

Table 4. Result of Scenario 2

Max Features.	Accuracy	F1-Score	Precision	Recall
500	86.51%	86.37%	86.33%	86.22%
1000	86.25%	86.28%	86.28%	86.50%
3000	86.98%	86.95%	86.92%	86.97%
5000	86.78%	86.86%	86.98%	86.76%
7500	86.31%	86.30%	86.29%	86.28%
10000	**87.57%**	**87.55%**	**87.52%**	**87.56%**
15000	87.04%	87.03%	87.01%	87.04%

Upon re-evaluation with cross-validation to ensure more reliable results, the performance metrics for different max features are presented in Table 5.

Table 5. Result of Scenario 2 using Cross Validation

Max Features.	Accuracy	F1-Score	Precision	Recall
500	86.15%	86.20%	86.30%	86.14%
1000	85.73%	85.83%	85.99%	85.80%
3000	86.84%	86.81%	86.83%	86.85%
5000	86.90%	86.79%	86.76%	86.86%
7500	86.86%	86.74%	86.69%	86.79%
10000	**86.96%**	**86.89%**	**86.88%**	**86.98%**
15000	86.86%	86.75%	86.71%	86.84%

By incorporating cross-validation, the model's performance was evaluated more robustly, ensuring that the reported metrics are reliable and not a result of overfitting to a particular train-test split. Based on **Table 4** and **Table 5**. Adjusting the max features parameter played a pivotal role in enhancing the model's performance. Specifically, setting the max features to 10,000 yielded the highest accuracy and F1-score. This suggests that a larger feature space provided the model with a more nuanced understanding of the text, capturing subtle variations in the language used in the reviews. The ability of TF-IDF to weigh words based on their importance across the dataset helped in filtering out noise and focusing on significant terms that contribute to the classification task. The superior performance with 10,000 features also highlights the importance of balancing the dimensionality of the feature space to avoid overfitting while ensuring sufficient representation of the data [19].

3.2.3 Third Scenario

The third scenario will utilize the Word2Vec Feature expansion and after implementing it while using the previous methods, the results of this scenario will be presented in Table 6.

Table 6. Result of Scenario 3

Accuracy	F1-Score	Precision	Recall
88.17%	88.13%	88.09%	88.14%

Based from the results in Table 6 it has been showed that the added method is improving our previous scenario models, the integration of Word2Vec for feature expansion significantly enhanced the model's capability to understand the semantic context of words [20]. The increase in accuracy and F1-score underscores the effectiveness of Word2Vec in capturing word relationships that are crucial for hate speech detection. By representing words as continuous vectors in a high-dimensional space, Word2Vec facilitated the model's ability to recognize patterns that are not immediately apparent from individual words. This method's ability to embed semantic context allowed the RNN to better capture the nuances of language used in hate speech, leading to a more robust classification performance [21]. The modest but notable improvement suggests that while Word2Vec contributes to enhancing the model, its effectiveness is maximized when combined with other advanced feature extraction and optimization techniques.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

3.2.4 Fourth Scenario

The fourth scenario will utilize the Differential Evolution Optimization model. The optimization process included a comparison between default learning rates and optimal learning rates. After implementing it into the previous model, the results of this scenario is presented in Table 7 the default learning rate means there's no optimization model used whereas optimized learning rate means there's optimization used.

Table 7. Result of Scenario 4

	Accuracy	F1-Score
Default Learning Rate	88.17%	88.13%
Optimal Learning Rate	88.63%	88.62%

The final scenario involved optimizing the learning rate of the RNN model using Differential Evolution. The results in Tabel 7 demonstrated that the optimized learning rate outperformed the default settings, achieving an accuracy of 88.63% and an F1-score of 88.62%. This improvement highlights the significance of fine-tuning hyperparameters to achieve optimal model performance. The Differential Evolution algorithm's ability to explore and exploit the search space effectively allowed for the discovery of a learning rate that balanced the trade-off between convergence speed and accuracy. The enhanced performance metrics indicate that the model was able to learn more efficiently from the training data, leading to better generalization on the test data. The optimization process proved to be a crucial step in refining the model, ensuring that it operates at its highest potential. All of the scenario graph will be depicted in Figure 4, 5, and 6 which illustrate the improvement over the last 4 scenario.

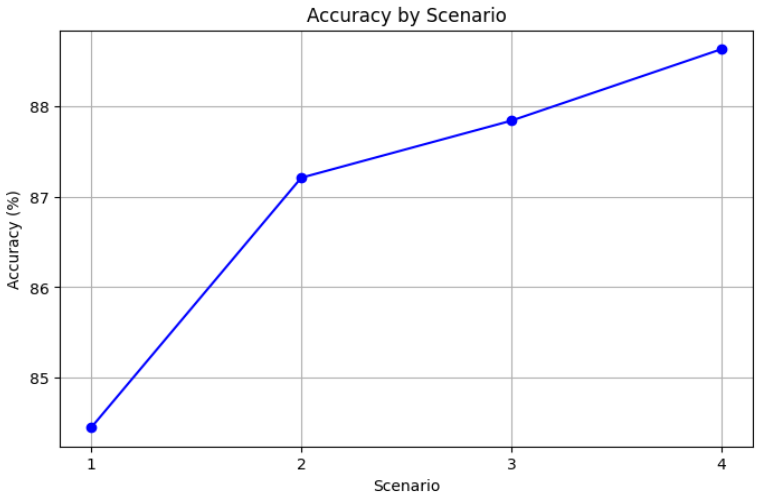

Figure 4. Accuracy of all scenario

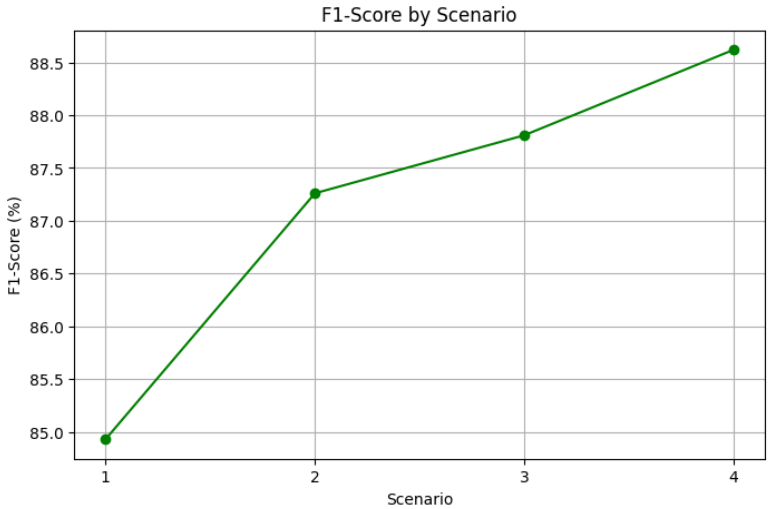

Figure 5. F1 Score of all scenario

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

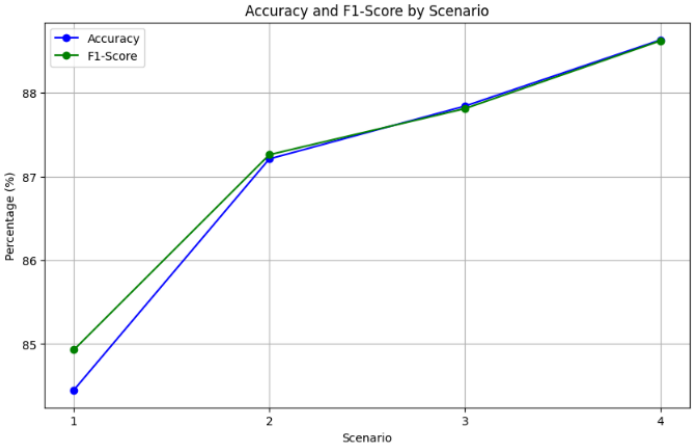

Figure 6. F1 Score and accuracy of all scenario

4. CONCLUSION

The research focused on improving hate speech classification in TikTok reviews using a dataset from the google play store. The analysis included classification through Recurrent Neural Network (RNN), feature extraction with TF-IDF, feature expansion using Word2Vec, and optimization using Differential Evolution Optimization. The results and findings are summarized as follows. Optimal Data Diversion the initial scenario tested different training and test data splits. The 90:10 split achieved the highest performance, with an accuracy of 84.45% and an F1 score of 84.93%. This split was used in subsequent scenarios to ensure consistent and superior results. TF-IDF Feature Extraction by adjusting the max features in TF-IDF, the model's performance was optimized. The best results were obtained with a max feature setting of 10,000, yielding an accuracy of 87.21% and an F1 score of 87.26%. This indicates that increasing the number of features significantly enhances the model's ability to classify hate speech. The implementation of Word2Vec for feature expansion further improved the model's performance, increasing the accuracy to 87.84% and the F1 score to 87.81%. This demonstrates the effectiveness of Word2Vec in enriching the feature set and improving classification outcomes. Differential Evolution Optimization in the final scenario, Differential Evolution Optimization was applied to determine the optimal learning rate. This optimization led to the highest performance metrics observed in the study, with an accuracy of 88.63% and an F1-score of 88.62%. This scenario confirmed that optimization techniques could further enhance the model's effectiveness. In conclusion, the iterative application of optimal data splits, TF-IDF feature extraction, Word2Vec feature expansion, and Differential Evolution Optimization collectively enhanced the performance of the RNN-based classification system. This research underscores the importance of feature optimization and iterative enhancements in developing robust machine learning models for hate speech detection. These findings provide a solid foundation for future studies and practical implementations in the realm of hate speech classification and making tiktok a safer place for the community.

REFERENCES

[1] J. W. Howard, “Free Speech and Hate Speech,” *Annu. Rev. Political Sci*, vol. 22, pp. 93–109, 2019, doi: 10.1146/annurev-polisci-051517.

[2] M. A. Fauzi and A. Yuniarti, “Ensemble method for indonesian twitter hate speech detection,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 11, no. 1, pp. 294–299, Jul. 2018, doi: 10.11591/ijeecs.v11.i1.pp294-299.

[3] R. Ahuja, A. Chug, S. Kohli, S. Gupta, and P. Ahuja, “The impact of features extraction on the sentiment analysis,” in *Procedia Computer Science*, Elsevier B.V., 2019, pp. 341–348. doi: 10.1016/j.procs.2019.05.008.

[4] K. Vijayaprabakaran and K. Sathiyamurthy, “Towards activation function search for long short-term model network: A differential evolution based approach,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 6, pp. 2637–2650, Jun. 2022, doi: 10.1016/j.jksuci.2020.04.015.

[5] A. Patel and A. K. Tiwari, “Sentiment analysis by using recurrent neural network,” in *Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE)*, 2019. doi: http://dx.doi.org/10.2139/ssrn.3349572.

[6] E. Çano and M. Morisio, “Word Embeddings for Sentiment Analysis: A Comprehensive Empirical Survey,” Feb. 2019, [Online]. Available: http://arxiv.org/abs/1902.00753

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024 Page: 807−816
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5520

[7] Y. Wang, A. Sun, J. Han, Y. Liu, and X. Zhu, "Sentiment analysis by capsules," in *The Web Conference 2018 - Proceedings of the World Wide Web Conference, WWW 2018*, Association for Computing Machinery, Inc, Apr. 2018, pp. 1165–1174. doi: 10.1145/3178876.3186015.

[8] F. Neutatz, B. Chen, Y. Alkhatib, J. Ye, and Z. Abedjan, "Data Cleaning and AutoML: Would an Optimizer Choose to Clean?," *Datenbank-Spektrum*, vol. 22, no. 2, pp. 121–130, Jul. 2022, doi: 10.1007/s13222-022-00413-2.

[9] C. Tangmanee, "User Test on Text-Based CAPTCHA: A Letter Case Examination," *Journal of Applied Security Research*, vol. 13, no. 2, pp. 250–266, Apr. 2018, doi: 10.1080/19361610.2018.1422372.

[10] H. T. Y. Achsan, H. Suhartanto, W. C. Wibowo, D. A. Dewi, and K. Ismed, "Automatic Extraction of Indonesian Stopwords," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 2, 2023, doi: 10.14569/IJACSA.2023.0140221.

[11] L. Sun, G. Zhao, Y. Zheng, and Z. Wu, "Spectral–Spatial Feature Tokenization Transformer for Hyperspectral Image Classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–14, 2022, doi: 10.1109/TGRS.2022.3144158.

[12] W. A. Prabowo and F. Azizah, "ARJUNA) Managed by Ministry of Research, Technology, and Higher Education," *Accredited by National Journal Accreditation*, vol. 4, no. 6, pp. 1142–1148, 2020, [Online]. Available: http://jurnal.iaii.or.id

[13] P. F. Muhammad, R. Kusumaningrum, and A. Wibowo, "Sentiment Analysis Using Word2vec and Long Short-Term Memory (LSTM) for Indonesian Hotel Reviews," in *Procedia Computer Science*, Elsevier B.V., 2021, pp. 728–735. doi: 10.1016/j.procs.2021.01.061.

[14] D. Jatnika, M. A. Bijaksana, and A. A. Suryani, "Word2vec model analysis for semantic similarities in English words," in *Procedia Computer Science*, Elsevier B.V., 2019, pp. 160–167. doi: 10.1016/j.procs.2019.08.153.

[15] M. A. Fauzi, "Word2Vec model for sentiment analysis of product reviews in Indonesian language," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 1, p. 525, Feb. 2019, doi: 10.11591/ijece.v9i1.pp525-530.

[16] P. Cen, K. Zhang, and D. Zheng, "Sentiment Analysis Using Deep Learning Approach," *Journal on Artificial Intelligence*, vol. 2, no. 1, pp. 17–27, 2020, doi: 10.32604/jai.2020.010132.

[17] A. Dahou, M. A. Elaziz, J. Zhou, and S. Xiong, "Arabic Sentiment Classification Using Convolutional Neural Network and Differential Evolution Algorithm," *Comput Intell Neurosci*, vol. 2019, 2019, doi: 10.1155/2019/2537689.

[18] Y. Yu, X. Si, C. Hu, and J. Zhang, "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures," *Neural Comput*, vol. 31, no. 7, pp. 1235–1270, Jul. 2019, doi: 10.1162/neco_a_01199.

[19] W. Shi, Y. Gong, C. Ding, Z. Ma, X. Tao, and N. Zheng, "Transductive Semi-Supervised Deep Learning Using Min-Max Features," 2018, pp. 311–327. doi: 10.1007/978-3-030-01228-1_19.

[20] Jack Merullo, Carsten Eickhoff, and Ellie Pavlick, "Language Models Implement Simple Word2Vec-style Vector Arithmetic.," *North American Chapter of the Association for Computational Linguistics*, 2023.

[21] S. Xiao, S. Huang, Y. Lin, Y. Ye, and W. Zeng, "Let the Chart Spark: Embedding Semantic Context into Chart with Text-to-Image Generative Model," *IEEE Trans Vis Comput Graph*, pp. 1–11, 2023, doi: 10.1109/TVCG.2023.3326913.