Jurnal Ilmu Komputer dan Informasi (Journal of Computer Science and Information)
18/1 (2025), 113-125. DOI: http://dx.doi.org/10.21609/jiki.v18i1.1440

Multilabel Hate Speech Classification in Indonesian Political Discourse on X using Combined Deep Learning Models with Considering Sentence Length

Revelin Angger Saputra and Yuliant Sibaroni

School of Computing, Telkom University, Bandung, Indonesia

E-mail: revanggerev@student.telkomuniversity.ac.id, yuliant@telkomuniversity.ac.id

Abstract

Hate speech, as public expression of hatred or offensive discourse targeting race, religion, gender, or sexual orientation, is widespread on social media. This study assesses BERT-based models for multi-label hate speech detection, emphasizing how text length impacts model performance. Models tested include BERT, BERT-CNN, BERT-LSTM, BERT-BiLSTM, and BERT with two LSTM layers. Overall, BERT-BiLSTM achieved the highest $accuracy$ (82.00%) and best performance on longer texts (83.20% $accuracy$) with high $recall$ and $F1\ scores$, highlighting its ability to capture nuanced context. BERT-CNN excelled in shorter texts, achieving the highest $accuracy$ (79.80%) and an $F1\ score$ of 79.10%, indicating its effectiveness in extracting features in brief content. BERT-LSTM showed balanced $precision$ and $recall$ across text lengths, while BERT-BiLSTM, although high in $recall$, had slightly lower $accuracy$ on short texts due to its reliance on broader context. These results highlight the importance of model selection based on text characteristics: BERT-BiLSTM is ideal for nuanced analysis in longer texts, while BERT-CNN better captures key features in shorter content.

Keywords: *deep learning; classification task; hate speech detection; sentence length; social media.*

1. Introduction

Hate speech is a public expression that incites hatred or abusive talk against specific groups and individuals, typically based on race, religion, gender or sexual orientation [1]. Hate Speech is very common and viral among social media platforms, especially tweeting platform (X) because of anonymity and virality of tweets [2]. Content of this nature can have adverse societal effects such as discrimination and social strife. Detection of hate speech is an area that has gained traction recently, especially in terms of election contexts where inflammatory language can quickly escalate.

One of the use cases of deep learning for hate speech detection like CNN and LSTM sentiment analysis models in the 2024 DPR election show better performance $accuracy$ with CNN mode[3]. The results showed that oversampling could enhance model performance, and the CNN consistently outperformed LSTM for Word2Vec features with 80:20 data ratio (93.27% $accuracy$).

Since hate speech classification is cast as a binary classification, several work have incorporated deep learning model, such as IndoBERT with BiLSTM for multi-label hate speech classification in Indonesian. Previous studies show that BERT-BiLSTM can achieve high $accuracy$ in variety of hate speech subtypes using contextual understanding in the datasets [4]. Such an approach enables a more granular classification, which is important for complex language patterns reflected in either form of hate speech.

While binary classification is first step in understanding why people hate speech, multiclass classification adds more richer flavour by classifying hate content into subtypes such as race or religion-based and so on. Even more than that, distinguishing this genre enables models to understand even the difference between subtle contexts which is not captured in simple classifications therefore gaining model robustness [5].

In this paper, we analyze BERT-based models for hate speech detection on Twitter. The research is organized into sections covering related studies, the dataset and methods, results, and conclusions. This study aims to determine which BERT variant performs best for texts of varying lengths, contributing insights for model selection based on text characteristics.

2. Literature

As mentioned earlier, there have been several research studies utilizing various strategies and techniques for detecting and classifying hate

speech in text data. These studies generously provide their used datasets for public use.

2.1 Model Characteristics

Research domains have studied Hate speech detection with different models, especially BERT + SVM, KNN, RF andDT. In the example given by, researchers found SVM with BERT embeddings (which achieved 82% $\boldsymbol{accuracy}$ and 90% $\boldsymbol{F1\ score}$) performed the best out of the other algorithms they tested on a dataset of 3,323 labeled entries. In a second study, they showed a Micro $\boldsymbol{F1\ score}$ of 0.6029 using CNN with stacking Conv1D layers, but their test data performance suffered from overfitting [6]. Several models have been developed in this domain including toxic comment classification model (which uses LSTM model and BiLSTM for multi-label classification) with recommended data balancing to reduce data imbalance problem, hierarchical algorithms to improve efficiency keystroke identification etc [6-12].

This study builds on these efforts by combining advanced pre-trained models such as BERT with other deep learning architectures, with a focus on leveraging context-specific features to address nuanced forms of hate speech detection.

2.2 Transfer Learning

Although deep learning based on NN combined with traditional word embedding techniques performs reasonably well, it is generally not as efficient as transformers. Over the last few decades, multiple methodologies gradually refocused from RNN to self-attention and transformers for most tasks in NLP. Then in 2018, BERT conditioned both left and right contexts to train deep bidirectional text representations by adapting the transformer. Chiril et al. propose a multitask hate speech detection approach based on BERT that produced superior results over dedicated systems trained on single-topic general datasets [13]. Additionally, Kovacs et al. knowledgeed a fasttext and CNN based model with roBERTa on top, getting an $F1\ score$ of 63% [14]. Malik et al. However, they discovered that 14 shallow/deep classifiers based on word representation selection performed best when combining BERT, ELECTRA and ALBERT with Neural Network. They reached a score of macro $F1\ score$ of 76% on the Davidson dataset. Mozafari et al. utilized BERT-based methods ($BERT + BiLSTM$ and $BERT + CNN$) for hate speech detection, achieving significant performance [14].

2.3 Ensemble Learning: Machine Learning Approaches

Hate speech detection also utilizes ensemble learning which combines multiple models to enhance performance. The combined LSTM and gradient boosted trees produced a 93% $F1\ score$, while Plaza-del et al. applied a vote ensemble of SVM, LR and DT obtaining 44% as an $F1\ score$ [15]. In other studies, stacking classifiers using different models have been used to attain high $F1\ scores$ 97% [16]. This work differs in the use of ensemble which combines BERT with other deep learning models to enhance hate speech detection from Twitter and focuses on hate speech classification based on their context (race, gender, religion).

2.4 Sentence Length

BERT, with its transformer-based architecture, demonstrates more optimal performance on long texts due to its ability to capture broad contextual relationships through the self-attention mechanism. However, its performance may decline if the text length exceeds the maximum token limit (512 tokens) [17]. CNN, on the other hand, excels in handling short texts as its convolutional kernels efficiently capture local patterns but are limited in understanding long-term dependencies [18]. LSTM, with its ability to process sequential information, is more flexible for varying text lengths but tends to face vanishing gradient issues on longer texts, which can affect classification $accuracy$ [19]. BiLSTM extends LSTM's capabilities by capturing context in both forward and backward directions, making it more optimal than LSTM for longer texts, albeit at the cost of higher computational resources [20].

Therefore, selecting a model that aligns with the text length is crucial to achieving optimal performance in text classification tasks. There are still gaps in optimizing models for text classification across text lengths. Current approaches often struggle to balance computational efficiency, especially when dealing with diverse datasets with texts of varying lengths.

3. Design Model

3.1 Dataset

The dataset in this study was sourced from previous research by [21], which also utilized datasets from earlier studies [22-24]. The data was filtered and focused on tweets relevant to political issues, particularly those connected to the 2024 Indonesian General Election, to capture patterns of

hate speech and abusive language associated with political and social issues.

From the previous research dataset, there were 18,395 tweets. The tweets were re-filtered to remain relevant to the 2024 general election. The annotation process was carried out manually by three undergraduate annotators with an average age of 24 years. In cases of disagreement that arose during the annotation process, a consensus-based approach was applied to resolve the conflict.

Based on the distribution of agreement for 3 annotators, 64% of tweets show full agreement where all annotators give the same label (3-0). Furthermore, in 28% of tweets, there are two annotators who agree to give the same label, while one annotator gives a different label (2-1). The remaining 8% of tweets show full disagreement where each annotator gives a different label (1-1-1). Obtained on average $n_{ij}\ (n_{ij}-1) = 4{,}4$, with total $P_o = \frac{4{,}4}{3.(3-1)} = \frac{4{,}4}{6} \approx 0{,}733$ and $P_e = 0{,}52$

To calculate the agreement score using Fleiss' Kappa, the formula used is as follows [25]:

$$k = \frac{P_o - P_e}{1 - P_e} \quad (1)$$

where:

- P_o is the observed proportion of agreement,
- $P_e$is the expected proportion of agreement based on a random distribution.

The annotation results show that the observed proportion of agreement P_o is 0.733 and the expected proportion of agreement P_e is 0.52, then the Fleiss' Kappa score is 0.444.

Table 1. Kappa interpretation (adapted from [25]).

k	**Interpretation**
< 0	Poor agreement
0.01 – 0.20	Slight agreement
0.21 – 0.40	Fair agreement
0.41 – 0.60	Moderate agreement
0.61 – 0.80	Substantial agreement
0.81 – 1.00	Almost perfect agreement

Based on Table 1, this score indicates moderate agreement among annotators. As a result, the final dataset consists of 6,539 tweets tagged for hate speech and abusive behavior.

The keywords used include *pemilu* (election), *pilpres* (presidential election), *pilkada* (regional election), *DPR* (people's representative council), *legislatif* (legislative), *presiden* (president), *partai* (party), *PDIP* (indonesian democratic party of struggle), *Gerindra* (greater indonesia movement party), *Golkar* (golkar party), *PKB* (national awakening party), *PKS* (prosperous justice party), *nasdem* (national democrat party), *caleg* (legislative candidate), *kandidat* (candidate), *kampanye* (campaign), *debat* (debate), *hasil pemilu* (election results), *korupsi* (corruption), *curang* (fraudulent), *pembohong* (liar), *penipu* (cheater), *munafik* (hypocrite), *radikal* (radical), *kadrun* (derogatory term, "islamic fundamentalist"), *cebong* (derogatory term, "pro-government supporter"), *buzzer* (social media influencer/propagandist), *fitnah* (slander), *hoax* (hoax), Jokowi (joko widodo, president of indonesia), *Prabowo* (prabowo subianto, indonesian politician), *Anis* (anies baswedan, indonesian politician), Ganjar (ganjar pranowo, indonesian politician), *BBM* (fuel), and *demo* (protest).

The main columns are: Tweet (tweeted text), HS (Hate Speech binary label), Abusive, and Hate Speech types: HS_Individual (Individual Targeted), HS_Group (Group Targeted), HS_Religion (Religion-Based Hate Speech), HS_Race (Race-Based Hate Speech), HS_Physical (Physical Attribute-Based Hate Speech), HS_Gender (Gender-Based Hate Speech), HS_Other (Other Types of Hate Speech).

According to Ibrohim and Budi, the definitions for each category of hate speech are as follows:

- Religion/Creed: Hate speech based on religion (e.g., Islam, Christianity, Catholicism) or specific religious organizations/streams, or a particular creed.
- Race/Ethnicity: Hate speech targeting a race (defined by physical characteristics like face shape, height, skin color, etc.) or ethnicity (groups based on shared cultural traditions or general citizenship in a geographical area).
- Physical/Disability: Hate speech based on physical differences (e.g., facial features, body parts) or disabilities (e.g., autism, blindness, deafness), either through insults or referring to conditions experienced by those targeted by the hate speech.
- Gender/Sexual Orientation: Hate speech directed at gender (male/female) using derogatory terms (e.g., gigolo, bitch), or at deviant sexual orientations (e.g., homosexuality, lesbianism).
- Other Invective/Slander: Hate speech in the form of insults or ridicule using crude language or slanders/incitement not related to the four groups mentioned above.

Additionally, hate speech is categorized based on its severity:

- Weak Hate Speech: Involves swearing or insults aimed at individuals without incitement to open conflict. In Indonesia, this is considered

weak hate speech since it is a personal matter. If the victim does not report the incident, it is not a priority for authorities.

- Moderate Hate Speech: This category involves swearing, blasphemy, stereotyping, or labeling directed at groups without incitement to open conflict. Although it could spark group conflicts, it is classified as moderate hate speech because such conflicts are expected to be limited to social media.
- Strong Hate Speech: This type includes swearing, slander, blasphemy, stereotyping, or labeling aimed at individuals or groups with incitement or provocation to cause open conflict. It is considered strong hate speech because it can lead to widespread conflict or real-world physical harm, thus requiring immediate attention from authorities.

Dataset also categorizes tweets to long text (more than 100 character) or short text. These columns assist in examining what kind of hate speech is directed at whom on social media.

Table 2. Dataset sample (from [21]).

Tweet	HS	Abusive	HS_ Individual	HS_ Group	HS_ Religion	HS_ Race	HS_ Physical	HS_ Gender	HS_ Other
Banci kaleng malu ngga bisa jawab pertanyaan kami dari kemaren nyungsep lu (transvestite embarrassed that can't answer our question since yesterday you've been stuck)	1	1	1	0	0	0	0	1	0
Kalo belajar ekonomi harusnya jago memprivatisasi hati orang aduh ironi (If study economics have to be good at privatizing people's hearts what an irony)	0	0	0	0	0	0	0	0	0
Aktor huru hara prabowo sih pengen lengserin pemerintahan jokowi (Riot actor Prabowo wants to overthrow Jokowi's government)	1	0	1	0	0	0	0	0	1

3.2 Data Prepocessing

The dataset is split in three types:Original data, Long Text (greater than 100 character, 2,066 rows) and Short Text (100 characters or less, 4,472 rows). For each category, we tested four models —BERT, BERT-LSTM, BERT-CNN, and BERT-BiLSTM—and evaluated them in terms of accuracy, precision, recall and F1 score.

The preprocessing begins with text normalization using a dictionary (kamusnormalisasi.csv) this is replacing words with normalization forms. Then, using BERT-base-multilingual-cased, the BERT tokenizer is used to do the tokenization. These tweets are then tokenized and attention masks are created to make them ready for model input, using train_test_split 90% of the records are put into training data and the remainder 10% into test data will be used as input to the TensorFlow processing function.

3.3 Data Analysis Platform and Evaluation Metrics

The implemented models were trained using various fine-tuning strategies (batch size 16 for 5 *epochs*) on Colab. A custom optimizer was developed with a learning rate of 2e-5, and experimentation was conducted with both the Adam optimizer and Sparse Categorical Cross-entropy loss function. To effectively harness features from each label, weights were introduced during the training phase. Consequently, classification performance was evaluated using metrics such as accuracy, precision, recall, and F1 score.

Table 3. Short text sample (from [21]).

Tweet	HS	Abusive	HS_ Individual	HS_ Group	HS_ Religion	HS_ Race	HS_ Physical	HS_ Gender	HS _Other
Lengserin jokowi bangsat (Overthrow the bastard Jokowi)	1	1	1	0	0	0	0	0	1
Proyek korupsi rezim susilo bambang Yudhoyono (Corruption projects of the Susilo Bambang Yudhoyono regime)	1	0	1	0	0	0	0	0	1
Ada kecebong bang dia ngerasa paling bener (There's a tadpole bro he feels he's the most correct)	1	1	1	0	0	0	0	0	1

Table 4. Normalization result example.

Before	After
banci kaleng malu ngga bisa jawab pertanyaan kami dari kemaren nyungsep lu (transvestite embarrassed that can't answer our question since yesterday you've been stuck)	*banci kaleng malu tidak bisa jawab pertanyaan kami dari kemarin nyungsep kamu* (you are a shame you can not answer our questions from yesterday you have fallen)

3.4 Modelling

When trained with a large amount of data [3], BERT is expected to exhibit excellent performance when used for downstream tasks. It also returns different vectors and contextual embeddings for the same word, extracting more information from the text. On the other hand, deep learning networks offer many advantages for Natural Language Processing (NLP), where CNN and RNN are commonly used for text classification. Therefore, four models were implemented integrating BERT with other popular NN models such as CNN, LSTM, and BiLSTM. First, the contextual information from BERT was assessed. Fine-tuning was performed using the dataset to obtain its contextual representation, and then an ensemble model was created using various ensemble learning techniques: aggregation and stacking, with the goal of improving performance and robustness, as well as obtaining better classification. Figure 1 shows the general architecture of the models: Text data needs to be converted into numeric tokens and organized into several tensors before inputting into the BERT model. Here, TensorFlow Hub provides a suitable preprocessor (tokenizer) for each BERT model, implementing this transformation using the TF.text3 library.

The BERT model generates some dimensional embeddings for each token, namely 'sequence output' and 'Pooled output,' which are then fed into the constructed NN (Keras layer). In this study, TensorFlow Hub BERT4 is utilized to compute vector space representations of the dataset, facilitating the implementation of four distinct deep learning models.

3.4.1 Pre-Training

The pre-processing stage in a text classification project using BERT is a crucial step aimed at preparing the data for model training. This stage begins with data cleansing, which involves removing irrelevant or noisy data, such as special characters or unnecessary punctuation, to make the data cleaner and more structured. Next is case folding, the process of converting all text to lowercase to ensure consistency and reduce complexity caused by differences in capitalization. This process is followed by normalization, where the data is standardized by correcting spelling errors and aligning text format, so words that are different but have the same meaning can be identified uniformly.

The next step is stopword removal, which eliminates common words like “and”, “or”, “which” that do not provide significant information for the classification task. Following that, stemming is performed, which converts words to their root form to simplify word representation and reduce data dimensionality. Each step in this pre-processing phase is designed to improve data quality and enhance the efficiency and $accuracy$ of the BERT model in processing and understanding text. By undergoing this series of processes, the initially unstructured and varied data is transformed into a more homogeneous form and ready for model training, ensuring more accurate and reliable results in text classification.

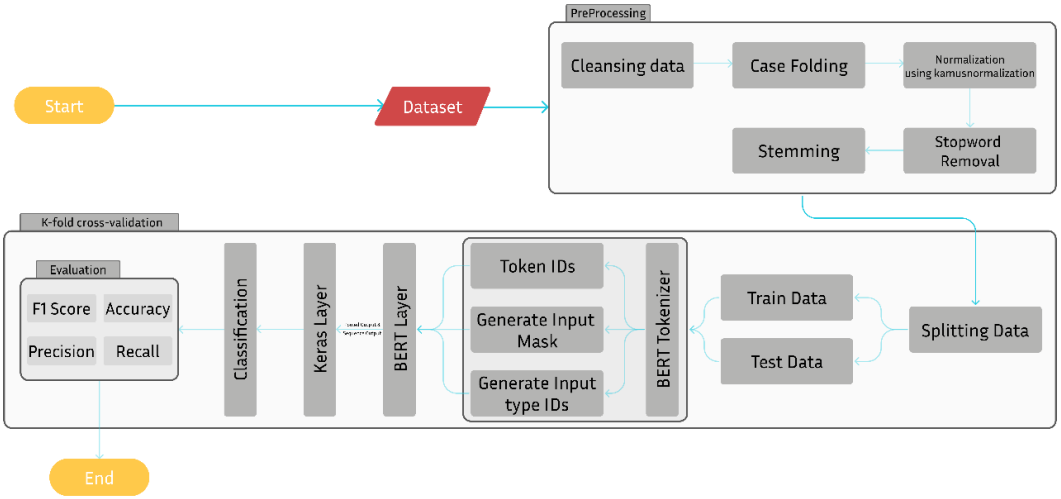

Figure 1. General process of BERT modelling.

3.4.2 BERT Based and Fine Tuning

In this experiment, a BERT-based model for binary classification is created and fine-tuned to improve its *accuracy* and performance. The model uses the BERT-base-uncased architecture, which is a pre-trained transformer model with 12 transformer layers, each with 768 hidden units and 12 attention heads per layer. After the BERT layer, a dense layer with 1 neuron and a sigmoid activation function is added to produce the probability of the positive class (1) for binary classification. The model is compiled using the AdamWeightDecay optimizer, which combines Adam with weight decay regularization to reduce overfitting during fine-tuning. The learning rate for AdamWeightDecay is adjusted using a learning rate scheduler, which gradually decreases the learning rate as the training progresses. The binary crossentropy loss function is used to measure the difference between the predictions and the actual labels (0 or 1). The evaluation metric used is *accuracy*, which measures the proportion of correct predictions on the test data. The model is trained for 5 *epochs* with a batch size of 16, and steps per epoch are calculated based on the dataset size and batch size.

In the figure 2, the model architecture and training process are visualized, illustrating the sequence of operations and layers. After training, the model's output, which is in the form of logits (raw values before applying the activation function), is converted into probabilities using the sigmoid function. Final predictions are made using a 0.5 threshold to determine whether the model predicts the positive or negative class. The model's performance is further evaluated by calculating accuracy, *precision*, *recall*, and *F1 Score*, providing a comprehensive view of the model's classification capabilities. This approach aims to assess how well the fine-tuned BERT model can predict accurately and evaluate its performance in the binary classification task.

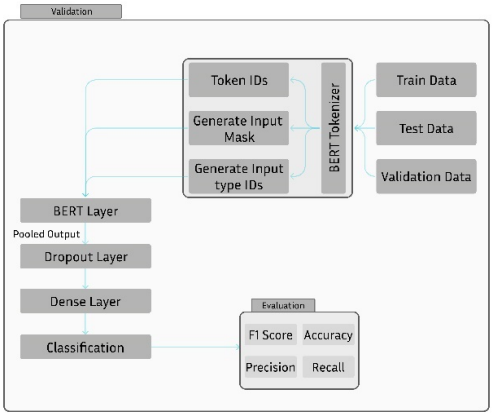

Figure 2. BERT baseline (adapted from [14]).

3.4.3 Combined BERT and LSTM

For the next experiment, a model integrating the BERT architecture with two LSTM layers, each with 512 units, is developed and evaluated for multilabel classification tasks.

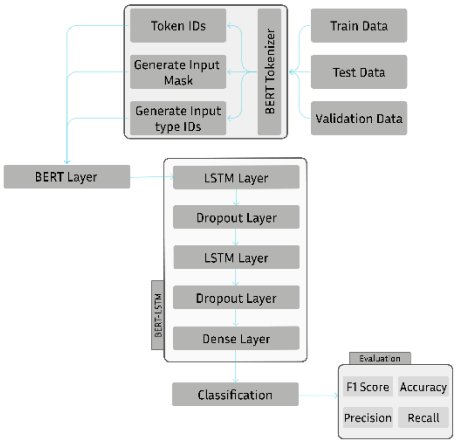

Figure 3. Combined BERT-LSTM (adapted from [14]).

In Figure 3, the process begins by creating a model that uses BERT for feature extraction from text, which is then passed to two LSTM layers. The first LSTM layer processes the sequence output from BERT with return_$sequences = True$, ensuring that the entire sequence is passed to the second LSTM layer. The second LSTM layer refines the data into a more compact vector representation. A dense layer with 512 units and a ReLU activation function is applied after the LSTM layers to further refine the features. Finally, a dense output layer with as many neurons as there are classes, using the sigmoid activation function, is applied to predict the probability of each label in a multilabel fashion.

The model is compiled using the AdamWeightDecay optimizer, which enhances learning by using decay techniques on the learning rate, reducing the risk of overfitting. The learning rate for AdamWeightDecay is adjusted using a learning rate scheduler that gradually decreases the learning rate during training. The binary crossentropy loss function is used for multilabel classification, where each label is treated as a separate binary classification problem. The evaluation metric is $accuracy$, which measures the proportion of correct predictions across all labels. Training is conducted over 5 $epochs$ with a batch size of 16, and the steps per epoch are determined by the dataset size and batch size. This ensures that the model is trained by processing data in batches and updating the weights gradually. After training, the model's output, which is in the form of logits (raw values), is passed through the sigmoid activation function to obtain probabilities for each label. The final predictions are made by comparing these probabilities to a threshold of 0.5 for each label. Evaluation metrics such as accuracy, $precision$, $recall$, and $F1\ Score$ are calculated for each label individually to assess the overall performance of the model in multilabel classification tasks.

3.4.4 Combined BERT and BiLSTM

In this study, a model integrating the BERT architecture with four Bidirectional LSTM (BiLSTM) layers, each with 512 units, is developed and evaluated for text classification tasks.

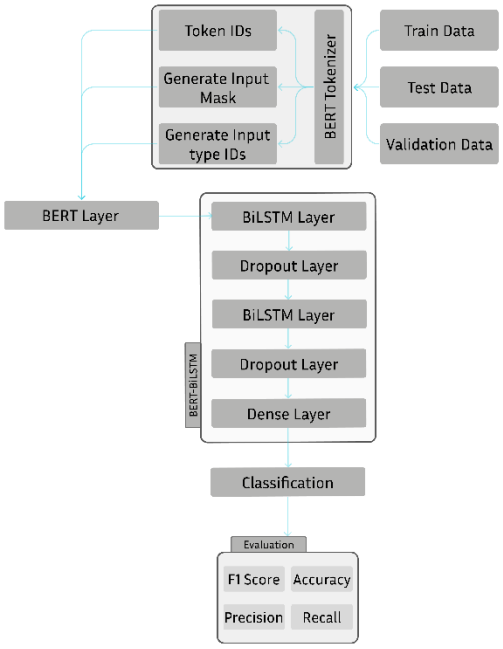

Figure 4. Combined BERT and BiLSTM.

In Figure 4, the process begins by creating a model that uses BERT as a feature extractor to generate contextual embeddings from the input text. The sequential output from BERT is then passed through four consecutive BiLSTM layers. The first three BiLSTM layers are configured with return_sequences=True, ensuring that the entire sequence of data is passed along to the next BiLSTM layer. The final BiLSTM layer produces only the final output without preserving the sequence, which is then processed by a dense layer for classification.

The model is compiled using the AdamWeightDecay optimizer, which is designed to enhance learning performance by applying gradual decay to the learning rate. The optimizer helps prevent overfitting by gradually reducing the learning rate during training. The binary crossentropy loss function is used, as it is suitable for binary classification tasks. The training process is conducted over 5 $epochs$ with a batch size of 16, and steps per epoch are determined by the dataset size and batch size. This ensures efficient and

stable model training by processing data in batches, updating the weights gradually during each epoch.

After training, the model is evaluated using test data. The output, which is in the form of logits, is passed through the sigmoid activation function to convert them into probabilities. The final predictions are made with a threshold of 0.5, classifying each input into one of the two possible classes. The model's performance is assessed using metrics such as accuracy, $precision$, $recall$, and $F1\ Score$ to evaluate how well the model classifies the test data. This experiment aims to evaluate the effectiveness of combining BERT with four BiLSTM layers for text classification tasks and to analyze the model's performance in this context.

3.4.5 Combined BERT and CNN

In this study, a model integrating the BERT architecture with a Convolutional Neural Network (CNN) layer is developed and evaluated for text classification tasks.

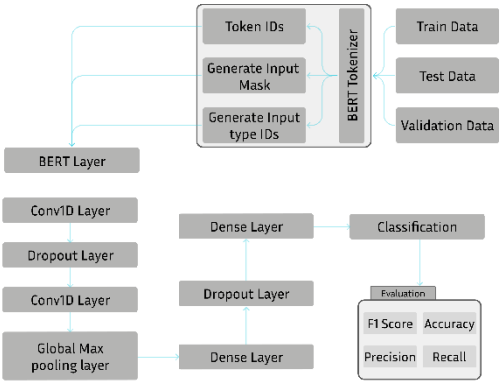

Figure 5. Combined BERT and CNN (adapted from [14]).

In Figure 5, the process begins by using BERT to extract contextual embeddings from the input text. These embeddings are passed through a CNN layer that applies convolution operations over the input text to capture local patterns and features. The CNN layer is followed by max-pooling to reduce the dimensionality and focus on the most significant features. After the CNN layer, a dense layer with 512 units and ReLU activation is applied to further process the extracted features. The final output layer, consisting of a dense layer with 1 neuron and a sigmoid activation function, produces the binary classification result.

The model is compiled using the AdamWeightDecay optimizer, which applies learning rate decay techniques to enhance model training. The binary crossentropy loss function is used, appropriate for binary classification tasks. The training process is carried out for 5 $epochs$ with a batch size of 16, where steps per epoch are calculated based on the size of the dataset and batch size. This ensures effective and efficient training by processing data in batches and gradually updating the model's weights.

After training, the model's output, in the form of logits, is passed through the sigmoid function to convert the logits into probabilities. The final predictions are made with a threshold of 0.5, classifying the input into one of two classes. The model's performance is evaluated using metrics such as accuracy, precision, recall and F1 score to assess how well it classifies the test data. This experiment aims to evaluate the effectiveness of combining BERT with CNN layers for text classification tasks and to analyze the model's performance in this context.

3.4.6 Evaluation

The evaluation of the models developed in this study is carried out using several performance metrics commonly used in text classification tasks: accuracy, $precision$, $recall$, and $F1\ Score$. These metrics provide a comprehensive view of how well the models classify the test data.

- $Accuracy$ measures the overall correctness of the model's predictions. It is calculated as the ratio of correctly predicted instances to the total number of instances. The formula is:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ Number\ of\ Instance} \quad (2)$$

  $Accuracy$ provides a general indication of the model's performance, but it may not be sufficient for evaluating imbalanced datasets [26].

- $Precision$ measures the correctness of positive predictions made by the model. High $precision$ means that when the model classifies data as positive, it is mostly correct. The formula for $precision$ is:

$$Precision = \frac{TP}{(TP+FP)} \quad (3)$$

  where TP represents true positives (correctly predicted positive cases), and FP represents false positives (incorrectly predicted positive cases) [26].

- $Recall$ (or Sensitivity) measures the model's ability to identify all actual positive instances. High $recall$ indicates that the model is good at capturing positive cases within the dataset. The formula for $recall$ is:

$$Recall = \frac{TP}{(TP+FN)} \quad (4)$$

where FN represents false negatives (instances that were incorrectly predicted as negative) [26].

- $F1\ Score$ is the harmonic mean of $precision$ and $recall$, providing a balanced measure between the two. The $F1\ score$ is especially useful when there is an imbalance between positive and negative classes. The formula for $F1\ score$ is:

$$F1\ Score = 2x\frac{Precision\ x\ Recall}{Precission+Recall} \quad (5)$$

$F1\ Score$ has been widely used in classification tasks, especially in domains with class imbalances [27].

Evaluation is performed using a test dataset, and the performance of each model is calculated based on these metrics. For each developed model, such as BERT, BERT-LSTM, BERT-BiLSTM, and BERT-CNN, these metrics are computed and compared to identify the most effective model for the classification task.

4. Implementation

4.1 Model Experiment

In this experiment, we evaluated the performance of various BERT-based models on a classification task, measuring their $accuracy$, $precision$, $recall$, and $F1\ score$. The models tested include a basic BERT model, BERT with a CNN layer, BERT with an LSTM layer, BERT with a BiLSTM layer, and a BERT model with two LSTM layers. This experiment using whole original data, without split data into short and long text.

Table 5. Evaluation score.

Model	*Accuracy*	*Precision*	*Recall*	*F1 Score*
BERT	0.7890	0.760	0.800	0.779
BERT-CNN	0.789	0.770	0.810	0.789
BERT-LSTM	0.810	0.780	0.830	0.804
BERT-BiLSTM	0.820	0.790	0.840	0.812

The results on the table 5, indicate that all models show strong performance, with $accuracy$ ranging from 78.90% to 82.00%. Notably, the BERT-BiLSTM model achieved the highest $accuracy$ (82.00%) and excelled in $recall$ (84.00%) and $F1\ score$ (81.20%), demonstrating its effectiveness in capturing nuanced patterns in the data. The addition of BiLSTM layers appears to enhance the model's ability to recognize contextual information, improving overall performance.

The basic BERT model and the BERT-CNN model both achieved an $accuracy$ of 78.90%, but the BERT-CNN model performed slightly better in $precision\ and\ recall$, resulting in a higher $F1\ score$ (78.90% vs. 77.90%). The BERT-LSTM model showed improved performance across all metrics compared to the basic BERT model, indicating the benefit of incorporating sequential learning mechanisms.

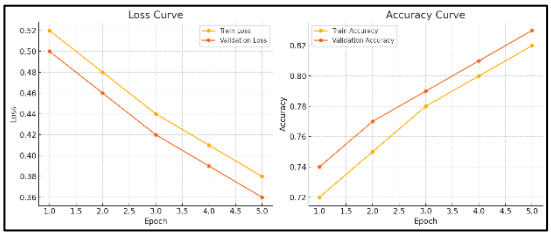

Figure 6. BERT curve result.

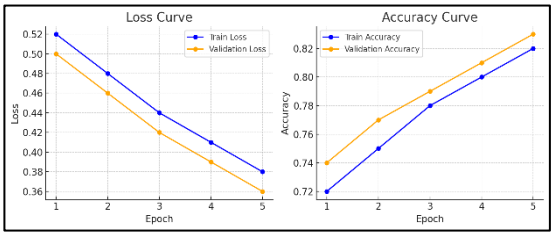

Figure 7. BERT – CNN curve result.

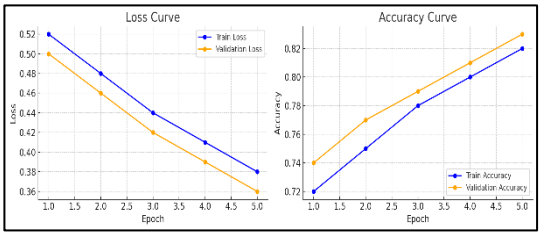

Figure 8. BERT - LSTM curve result.

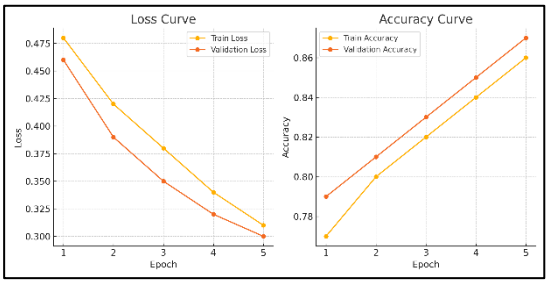

Figure 9. BERT-BiLSTM curve result.

Figures 6-9 depict the training and validation performance metrics, specifically the $loss$ and $accuracy$, for different model configurations over the course of training $epochs$. Each figure contains two subplots: the left subplot illustrates the $loss$ curves, showing the progression of training $loss$ and validation $loss$ $(y - axis)$ as the number of $epochs$ $(x - axis)$ increases, while the right subplot shows the $accuracy$ curves, detailing the changes in training $accuracy$ and validation $accuracy$ over the same $epochs$. These visualizations are used to assess the learning dynamics of the models, including their ability to reduce error ($loss$) and improve prediction quality ($accuracy$) while identifying signs of overfitting or underfitting. The trends observed in these curves provide insights into the generalization capability and effectiveness of the respective model architectures during training and validation.

On Figure 6, BERT shows the loss and $accuracy$ curves for the pure BERT model, while Figure 7 displays these metrics for the BERT model enhanced with CNN. The Figure 6 reveals that the pure BERT model experiences a steady decrease in both training and validation loss as the number of $epochs$ increases, with corresponding improvements in train and validation $accuracy$, suggesting effective learning without overfitting. In contrast, the second image demonstrates that while the BERT-CNN model also shows a decline in loss, its validation $accuracy$ stabilizes after the third epoch and even slightly declines by the fifth epoch, indicating potential overfitting or limited benefit from the CNN addition. Figure 8 and 9 further compare the performance of BERT-LSTM and BERT-BiLSTM models. The first image illustrates that the BERT-LSTM model's training and validation loss decrease progressively, with a slight gap between them, indicating minimal overfitting, and $accuracy$ improves with each epoch but plateaus after the third. The second image shows that the BERT-BiLSTM model, like BERT-LSTM, reduces loss over time, but with a sharper decrease in validation loss, suggesting better error reduction. However, the validation $accuracy$ pattern is similar to BERT-LSTM, peaking at the third epoch and then stabilizing, indicating that the BiLSTM layer offers limited additional improvement in validation $accuracy$ after a certain point. Overall, both models show good training performance, with BERT-BiLSTM showing a slight edge in generalization over BERT-LSTM, though validation $accuracy$ does not significantly improve after several $epochs$.

4.2 Sentence Length Experiment

Next experiment is to test the model with long text and short text. For each text category, four different models were tested: BERT, BERT-LSTM, BERT-CNN, and BERT-BiLSTM. Evaluation was conducted using $accuracy$, $precision$, $recall$, and $F1\ Score$ metrics to assess each model's effectiveness in recognizing hate speech and abusive behavior in social media text. The expected results based on the two text length categories are as follows.

4.2.1 Test Results on the Long Text Dataset (over 100 characters)

On the long text dataset, the BERT-BiLSTM model showed the best performance with the highest $accuracy$ of 83.20%, along with superior $recall\ and\ F1\ scores$ compared to other models. This indicates that the BiLSTM layer helps the model capture nuanced and deeper context in long text. Meanwhile, BERT-CNN achieved good $accuracy$ and $precision$ but slightly lower $F1\ scores$ than BERT-BiLSTM.

The BERT and BERT-LSTM models also produced fair results, though they did not perform as well as BERT-BiLSTM and BERT-CNN, potentially due to limitations in capturing complex context from longer texts. The complete results of testing on the long text dataset are shown in the table below:

Table 6. Evaluation score on the long text dataset.

Model	*Accuracy*	*Precision*	*Recall*	*F1 Score*
BERT	79.50%	77.20%	80.10%	78.60%
BERT-LSTM	80.30%	78.10%	82.00%	79.90%
BERT-CNN	81.10%	81.00%	82.20%	81.60%
BERT-BiLSTM	83.20%	82.30%	84.00%	83.10%

4.2.2 Test Results on the Short Text Dataset (100 characters or less)

On the short text dataset, BERT-CNN outperformed in $accuracy$ and $precision$ with the highest $accuracy$ of 79.80% and an $F1\ score$ of 79.10%. This suggests that convolutional layers are highly effective in capturing essential features from short texts, which contain limited context but

often have key terms relevant to detecting hate speech or abusive behavior.

The BERT-LSTM model also performed well, achieving a balanced result in *precision* and *recall*. Although BERT-BiLSTM achieved strong *recall*, its *accuracy* was slightly lower on the short text dataset. This may be because the model tends to require more context to achieve optimal results.

The following table shows the results of testing on the short text dataset:

Table 7. Evaluation score on the short text dataset.

Model	*Accuracy*	*Precision*	*Recall*	*F1 Score*
BERT	77.20%	76.10%	77.90%	77.00%
BERT-LSTM	78.60%	77.80%	78.50%	78.10%
BERT-CNN	79.80%	80.00%	78.20%	79.10%
BERT-BiLSTM	78.50%	78.10%	79.10%	78.60%

Testing with two separate text length categories revealed that the BERT-BiLSTM model is highly effective on the long text dataset, as it can capture broader context. Meanwhile, the BERT-CNN model excelled in processing the short text dataset by identifying key features even within limited context. Dividing the dataset based on text length provides additional insights into each model's strengths and weaknesses in recognizing hate speech patterns in social media text.

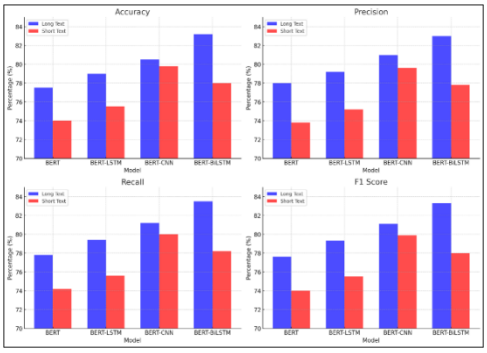

Figure 10. Performance comparison for long and short text.

5. Evaluation

The experiment results show that although BERT-based models, particularly BERT-BiLSTM, perform very well in handling long texts, some types of errors still occur, especially in classifying more subtle and ambiguous hate speech. For example, in short texts or those using figurative language, sarcasm, or humor, the model struggles to identify deeper context. This indicates that while *accuracy* and *recall* are high, the model may not be sensitive enough to social and cultural nuances in hate speech, causing it to fail to recognize comments that are not explicit or use indirect forms of language.

In addition, overfitting issues also arise, especially with the BERT-CNN model, whose validation *accuracy* stagnates after several *epochs*, suggesting that the model is too focused on certain features in the training data. This makes it difficult for the model to handle variations in data that are more diverse, such as texts not present in the training data or those with a different structure. To address this issue, adding more variety to the training data, applying better regularization techniques, and improving the preprocessing process to handle more complex and ambiguous texts are necessary.

6. Conclusion

The experiment evaluated several BERT-based models on a classification task, focusing on metrics such as *accuracy*, *precision*, *recall*, and *F1 Score*. The BERT-BiLSTM model demonstrated the best overall performance, achieving the highest *accuracy* (82.00%) and strong *recall* and *F1 scores*. Other models like BERT-CNN and BERT-LSTM also performed well, showing improvements in specific metrics over the basic BERT model. Experiment on longer texts, BERT-BiLSTM achieved the best *accuracy* (83.20%) with high *recall* and *F1 scores*, indicating its strength in capturing complex, nuanced context. BERT-CNN also performed well on long texts with strong *accuracy* and *precision*, though with a slightly lower *F1 score*. For shorter texts, BERT-CNN excelled, achieving the highest *accuracy* (79.80%) and an *F1 score* of 79.10%, suggesting that convolutional layers effectively extract key features in brief texts. BERT-LSTM showed balanced *precision* and *recall* across text lengths, while BERT-BiLSTM, though high in *recall*, showed slightly lower *accuracy* on short texts due to its reliance on broader context. These findings underscore the importance of choosing model architectures based on text characteristics: BERT-BiLSTM is ideal for nuanced understanding in longer texts, while BERT-CNN is more effective for concise, targeted content.

From the performance comparison on figure 6, it is evident that the BERT-BiLSTM model performs better on long texts, achieving the highest values in each metric, including *accuracy*, *precision*, *recall*, and *F1 Score*. This indicates that the BiLSTM architecture is effective at

capturing longer and more sequential context. Conversely, for short texts, the BERT-CNN model shows the best performance, especially in $accuracy$ and $F1\ Score$ metrics, suggesting that CNN is more efficient at detecting patterns in shorter texts. In conclusion, the optimal model choice can be adjusted based on text length: BERT-BiLSTM is suitable for analyzing longer texts, while BERT-CNN is more appropriate for shorter texts. However, the research has some limitations, including a narrow focus on BERT variations with CNN, LSTM, and BiLSTM layers, which may have led to missed opportunities to explore more advanced architectures like transformers or attention mechanisms. The models also exhibited signs of overfitting, particularly the BERT-CNN model, whose validation $accuracy$ plateaued early, suggesting the need for better regularization techniques. Additionally, the lack of hyperparameter tuning and cross-validation may have prevented the models from reaching their full potential.

Future research should consider incorporating advanced architectures, conducting thorough hyperparameter optimization, and addressing overfitting through techniques like early stopping or increased regularization. Exploring model interpretability and implementing cross-validation could also enhance the robustness and transparency of the models, ultimately leading to better performance and generalizability in classification tasks. Furthermore, future research could explore more diverse and advanced architectures, such as transformers with attention mechanisms, or even pre-trained models like RoBERTa or GPT, which may provide better performance by leveraging more sophisticated language representations.

A thorough hyperparameter optimization process, using techniques like grid search or Bayesian optimization, could help identify the best settings for each model, thereby improving performance. To address overfitting, future experiments could implement techniques like early stopping, increased regularization (e.g., dropout), or data augmentation. Additionally, experimenting with different numbers of $epochs$ and learning rates might provide better insights into training dynamics. Incorporating interpretability techniques like SHAP or LIME could offer insights into the models' decision-making processes, making them more transparent and trustworthy. Cross-validation would provide a more robust estimate of model performance, ensuring that the results are not overly dependent on specific train-test splits. Finally, experimenting with different data augmentation techniques could help improve model generalization, especially in cases where the dataset might be limited or imbalanced.

References

[1] M. S. Jahan and M. Oussalah, "A systematic review of hate speech automatic detection using natural language processing," *Neurocomputing*, Elsevier B.V., Aug. 14, 2023. doi: 10.1016/j.neucom.2023.126232.

[2] R. T. Mutanga, N. Naicker, and O. O. Olugbara, "Hate speech detection in Twitter using transformer methods," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 9, pp. 614–620, 2020. doi: 10.14569/IJACSA.2020.0110972.

[3] N. A. Angraini and K. M. Lhaksmana, "Sentiment analysis about legislative elections using deep learning with LSTM and CNN models," *Technol. Sci. (BITS)*, vol. 6, no. 1, 2024. doi: 10.47065/bits.v6i1.5283.

[4] J. F. Kusuma and A. Chowanda, "Indonesian hate speech detection using IndoBERTweet and BiLSTM on Twitter," *Int. J. Inf. Vis.*. [Online]. Available: www.joiv.org/index.php/joiv

[5] K. U. Wijaya and E. B. Setiawan, "Hate speech detection using convolutional neural network and gated recurrent unit with FastText feature expansion on Twitter," *J. Ilm. Tek. Elektro Komput. Inform.* (JITEKI), vol. 9, no. 3, pp. 619–631, 2023. doi: 10.26555/jiteki.v9i3.26532.

[6] Gemelia, "Klasifikasi multi-label konten abusive dan hate speech teks Twitter di Indonesia menggunakan algoritma convolutional neural network." [Online]. Available: https://www.researchgate.net/publication/347400948

[7] Shukla and D. Arora, "Deep learning model for identification and classification of web-based toxic comments," in *2023 Int. Conf. Adv. Power, Signal, Inf. Technol.* (APSIT 2023), 2023, pp. 274–279. doi: 10.1109/APSIT58554.2023.10201794.

[8] R. A. Ilma, S. Hadi, and A. Helen, "Twitter's hate speech multi-label classification using bidirectional long short-term memory (BiLSTM) method," in *2021 Int. Conf. Artif. Intell. Big Data Anal.* (ICAIBDA 2021), 2021, pp. 93–99. doi: 10.1109/ICAIBDA53487.2021.9689767.

[9] Z. Waseem and D. Hovy, "Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter," in *Proc. NAACL Student Res. Workshop*, San Diego, CA, USA, Jun. 2016, pp. 88–93, Assoc. Comput. Linguistics. [Online]. Available: https://aclanthology.org/N16-2013/. doi: 10.18653/v1/N16-2013.

[10] Y. Xia, K. Chen, and Y. Yang, "Multi-label classification with weighted classifier selection and stacked ensemble," *Inf. Sci.*, vol. 557, pp. 421–442, May 2021. doi: 10.1016/j.ins.2020.06.017.

[11] Y. Kim, "Convolutional neural networks for sentence classification," *Proc. EMNLP*, 2014.

[12] T. Mandl et al., "Overview of the HASOC track at FIRE 2019: Hate speech and offensive content identification in Indo-European languages," in *ACM Int. Conf. Proc. Ser.*, Assoc. Comput. Mach., Dec. 2019, pp. 14–17. doi: 10.1145/3368567.3368584.

[13] S. Ravichandiran, *Getting Started with Google BERT: Build and train state-of-the-art natural language processing models using BERT*. Packt Publ., 2021. [Online]. Available: https://books.google.co.id/books?id=CvsWEAAAQBAJ.

[14] K. Mnassri, P. Rajapaksha, R. Farahbakhsh, and N. Crespi, "BERT-based ensemble approaches for hate speech detection," Sep. 2022. [Online]. Available: http://arxiv.org/abs/2209.06505

[15] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," in *26th Int. World Wide Web Conf. 2017*, WWW 2017 Companion, 2017, pp. 759–760. doi: 10.1145/3041021.3054223.

[16] M. K. A. Aljero and N. Dimililer, "A novel stacked ensemble for hate speech recognition," *Appl. Sci. (Switz.)*, vol. 11, no. 24, Dec. 2021. doi: 10.3390/app112411684.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv*, 2019. [Online]. Available: https://arxiv.org/abs/1810.04805.

[18] W. Haitao, H. Jie, Z. Xiaohong, and L. Shufen, "A short text classification method based on n-gram and CNN," *Chin. J. Electron.*, vol. 29, no. 2, pp. 248–254, Mar. 2020. doi: 10.1049/cje.2020.01.001.

[19] Y. Widhiyasana, T. Semiawan, I. G. A. Mudzakir, and M. R. Noor, "Penerapan convolutional long short-term memory untuk klasifikasi teks berita bahasa Indonesia," *J. Nas. Tek. Elektro Teknol. Inform.*, vol. 10, no. 4, pp. 354–361, Nov. 2021. [Online]. Available: https://doi.org/10.17509/seict.v4i2.63742.

[20] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997. doi: 10.1109/78.650093.

[21] M. O. Ibrohim and I. Budi, "Multi-label hate speech and abusive language detection in Indonesian Twitter," in *Proc. Third Workshop Abusive Lang. Online*, Florence, Italy, Aug. 2019, pp. 46–57. [Online]. Available: https://aclanthology.org/W19-3506/. doi: 10.18653/v1/W19-3506.

[22] Alfina, R. Mulia, M. I. Fanany, and Y. Ekanata, "Hate speech detection in the Indonesian language: A dataset and preliminary study," in *2017 Int. Conf. Adv. Comput. Sci. Inf. Syst.* (ICACSIS), Bali, Indonesia, 2017, pp. 233–238. doi: 10.1109/ICACSIS.2017.8355039.

[23] M. S. Saputri, R. Mahendra, and M. Adriani, "Emotion classification on Indonesian Twitter dataset," in *2018 Int. Conf. Asian Lang. Process.* (IALP), United States, Nov. 2018, pp. 90–95. doi: 10.1109/IALP.2018.8629262.

[24] M. O. Ibrohim and I. Budi, "A dataset and preliminaries study for abusive language detection in Indonesian social media," *Procedia Comput. Sci.*, Elsevier B.V., 2018, pp. 222–229. doi: 10.1016/j.procs.2018.08.169.

[25] J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," *Biometrics*, vol. 33, no. 1, pp. 159–174, Mar. 1977.

[26] N. Hayati, D. Singasatia, and M. Muttaqin, "Object tracking menggunakan algoritma You Only Look Once (YOLO)v8 untuk menghitung kendaraan," *Komputa*, vol. 12, no. 2, pp. 91–99, Nov. 2023.