Adam and AdamW Optimization Algorithm Application on BERT Model for Hate Speech Detection on Twitter

1st Putri Nabila
School of Computing
Telkom University
Bandung, Indonesia
putrin@student.telkomuniversity.ac.id

2nd Erwin Budi Setiawan
School of Computing
Telkom University
Bandung, Indonesia
erwinbudisetiawan@telkomuniversity.ac.id

***Abstract*—Twitter is one of the social media platforms used to channel user's opinions and it is common for users to misuse this feature to express hate speech. Hate speech, intentional or unintentional, can trigger conflicts, especially on social media platforms with a large user base like Twitter, where hate speech can easily spread and reach its targets. Therefore, a system is needed to detect hate speech to prevent its spread and handle it. In this study, hyperparameter fine-tuning is performed on a pretrained BERT Model classify data containing hate speech with the collected data from Twitter. Adam Optimizer and AdamW Optimizer are used on the constructed BERT model, and their accuracies will be compared to determine which one yields the best result. This research found that the fine tuning the right parameter for each optimizer can result in a significantly higher accuracy compared to the model with default parameter setting. For the task of identifying hate speech in data, BERT model optimized by AdamW optimizer can reach an accuracy of 90.08% by setting the initial learning rate to 1e-5 and initial weight decay to 1e-3, an increase of 40.03% from the baseline. While BERT model optimized with Adam reaches 90.03% in accuracy by setting the initial learning rate to 1e-5 and using its default initial weight decay, an increase of 40.38% from the baseline.**

Keywords—Twitter, BERT, Optimizer, Adam, AdamW, Hate Speech

I. INTRODUCTION

Twitter is one of the social media platforms widely used by people around the world, including Indonesia. According to the Digital 2023 Global Overview Report, Twitter had a total of 556 million users as of January 2023, ranking 14th among the most widely used social media applications globally [1]. In Indonesia, there were 24 million Twitter users recorded in early 2023, placing it as the 5th country with the highest number of Twitter users worldwide, according to the Digital 2023 Indonesia report [2].

Users of Twitter express their opinions and communicate with others through their tweets. However, it is common for users to post tweets containing hate speech, whether intentionally or unintentionally. The nature of information openness on social media, where communication takes place publicly, is one of the triggers for the high tendency of society to engage in hate speech on social media platforms[11]. Twitter's policy on hate speech behavior, implemented in April 2023, states that both tweets and users violating this policy will face sanctions [3]. However, this policy has not completely prevented users from creating tweets containing hate speech. Additionally, Twitter still relies on user reports to detect and address hate speech tweets.

In Indonesia, there are already laws regulating hate speech on social media, such as Law No. 19 of 2016 on Amendments to Law No. 11 of 2008 concerning Electronic Information and Transactions (ITE Law) [4]. However, community assistance is still needed to report hate speech cases on social media, which are then analyzed physically as evidence. This indicates the need for an effective hate speech detection system on social media to better handle such cases.

Hate speech involves communication actions by individuals or groups in the form of provocation, insults, defamation, incitement, and spreading false information based on aspects such as race, gender, ethnicity, physical disability, sexual orientation, nationality, religion, and others [5]. Text considered hate speech often contains offensive language [19]. In Indonesia, offensive language is commonly derived from unfavorable conditions such as disabilities, mental illnesses, sexual deviations, lack of modernization, animal names, and others [20]. Hate speech cases can impact others and trigger conflicts. Information on social media spreads wide and reaches individuals or groups faster, especially on platforms with a large user base like Twitter. Therefore, a robust system is required to detect hate speech on social media, facilitating faster responses and preventing its spread.

Several studies have used BERT as a model for hate speech classification. Research by Saleh, Alhothali & Moria in 2023 demonstrated that BERT outperformed two out of three deep models combined with word embeddings [6]. Another study by Dowlagar & Mamidi in 2021 compared BERT with Support Vector Machine in hate speech detection, where BERT outperformed other models on all three tested datasets [7]. Hence, this research aims to train a BERT model with a classification layer for hate speech classification through fine-tuning.

Adam Optimizer is a stochastic optimization method that only requires first-order gradients with low memory requirements, primarily designed for neural network optimization [8]. AdamW Optimizer is a variant of Adam Optimizer equipped with weight decay regularization parameters. Both optimization methods are applied to the system to adjust parameters of the model, producing higher accuracy predictions. By fine-tuning the parameters such as learning rate and weight decay in both models with Adam and AdamW Optimizer, in hope of acquiring the best results in detecting hate speech in text.

The system is designed to assist in the effort to detect hate speech case faster and more effectively. With a reduction in hate speech that can provoke conflicts, social media can become a safer and more orderly environment for users.

2024 International Conference on Data Science and Its Applications (ICoDSA) | 979-8-3503-6535-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICODSA62899.2024.10651619

II. Related Works

A. Literature Review

With the increasing prevalence of hate speech cases, there has been a growing number of studies focusing on hate speech as their main topic. Several research efforts have addressed the classification of hate speech using various algorithms to assess the accuracy of the chosen algorithmic models. The classification of hate speech using fine-tuned BERT models has also been explored in several studies.

In the study by Saleh, Alhothali & Moria (2023) [6], a comparison was made between hate speech detection using BERT and Hate Speech Word Embedding with a Deep Model. For the first approach, a BiLSTM classifier was employed, combined with feature extraction methods such as Glove, Google Word2vec, and domain-specific (hate domain) word embedding. The second approach utilized fine-tuned BERT for hate speech classification. The comparison of their F1 scores revealed that, although domain-specific word embedding still outperformed, the BERT model succeeded in surpassing other models.

Another study by Dowlagar & Mamidi (2021) [7] compared BERT with Support Vector Machine and ELMO in hate speech detection. The research involving three datasets in English, German, and Hindi, tested on all three models. BERT outperformed both models in all tested datasets, achieving F1 scores of 54.44% for the English dataset, 47.78% for the German dataset, and 49.71% for the Hindi dataset.

A study conducted by Sun et al. [22] demonstrated how to fine-tune BERT for text classification, where one of the processes is to fine-tune the hyperparameters of the model. Learning rate are fine-tuned to prevent catastrophic forgetting, which is a phenomenon in transfer learning where the model loses previously learned information when trained with new data [23].

Based on the literature review, the author decided to use BERT model for hate speech classification with Adam and AdamW optimizer to optimize the model's parameters on hate-speech classification task. Fine-tuning the hyperparameters of Adam Optimizer and AdamW Optimizer, such as learning rate and weight decay, is done to adjust the model's parameters better to the task given and generate better predictions.

B. BERT

Bidirectional Encoder Representations from Transformers, or BERT, is a Transformer-based language representation model designed for pre-training with deep bidirectional representations of unlabeled text from BooksCorpus and Wikipedia [12]. BERT contains 12 Transformer blocks [21], 12 self-attention heads and 768 hidden layers. BERT training consists of two stages: pre-training and fine-tuning. The pre-training phase involves two unsupervised tasks: Masked Language Model (MLM) and Next Sentence Prediction (NSP). In MLM, some input tokens are masked, and the model predicts these hidden tokens. NSP trains the model to understand the relationship between two sentences. During the fine-tuning stage, input and output data corresponding to the specific task are provided to BERT to adjust its parameters. In this research, a pre-trained BERT model will be used to be fine-tuned in hate speech classification task.

C. Adam Optimizer

Adam Optimizer, or Adaptive Moment Optimizer, is a gradient-based optimization algorithm for the first order of a stochastic objective function based on adaptive estimates of lower-order moments (Kingma & Ba, 2017) [8]. This method is easy to implement, computationally efficient, has low memory requirements, and is suitable for large-scale data or parameter problems. Adam Optimizer combines two other optimization algorithms, namely the Momentum Optimizer and RMSprop (Root Mean Square Propagation) [13].

Adam Optimizer performs bias correction by calculating the corrected estimates using (1) and (2):

$$\widehat{m_t} = m_t \div (1 - \beta_1^t) \tag{1}$$

$$\widehat{v_t} = v_t \div (1 - \beta_2^t) \tag{2}$$

m_t represents the first-order moment estimate that moves the average of the gradients and v_t represents the second-order moment estimate that moves the average of squared gradients. While β is the exponential decay rate of each variable.

Gradient derivatives are then computed, and the parameters are updated whenever a change occurs. Substitute the old parameters with the updated ones to get the following equation:

$$w_t = w(t-1) - \alpha * \left(\left(\frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}}\right) + e\right) \tag{3}$$

In which w_t represents the weights of the neural network at a certain time step and α represents the learning rate. The update iterations will continue until the model converges.

D. AdamW Optimizer

The AdamW Optimizer is a variant of the Adam Optimizer that decouples weight decay from gradient based update. The update rule of AdamW are slightly differs from those of Adam Optimizer, where the weight decay are calculated separately from the L_2 regularization in SGD. This method is introduced to prevent weight decay issues that may occur in the Adam algorithm, ultimately leading to better generalization. [14]

Weight decay helps prevent overfitting by penalizing large weights during the training process. In the context of Adam, the original algorithm did not effectively handle weight decay, potentially leading to suboptimal generalization performance. AdamW addresses this limitation by introducing a dedicated weight decay term. This modification ensures that the weight decay regularization is applied correctly and independently of the other adaptive learning rate components, contributing to improved stability and generalization capabilities.

III. System Design

Below is the chart of the system designed for the research that will be conducted.

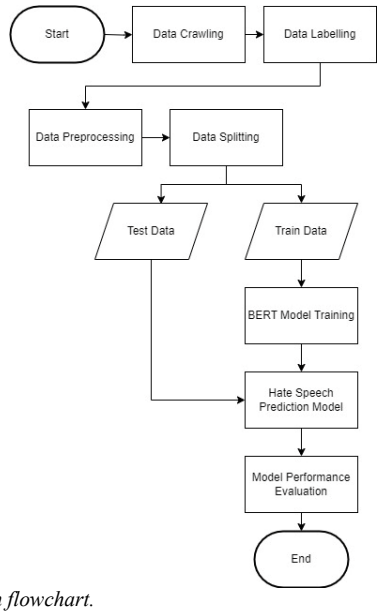

Fig. 1. System flowchart.

As shown in Fig. 1, the process begins with data crawling. The collected data will be assigned to its corresponding labels. After labelling, the data will then get preprocessed and split into train dataset and test dataset. A pre-trained BERT Model will be fine-tuned using the train dataset for hate speech recognition task. Adam and AdamW optimizer will be applied on the model in hope of achieving better accuracy. After fine-tuning, the model will be evaluated with confusion matrix using the test dataset.

A. Data Crawling

Data is obtained by crawling through Twitter for tweets containing keywords from trending topics in Bahasa. Data are collected from August 2023 to December 2023. Datasets from Jonathan [16] and Hidayat [17] are also used in addition to balance the ratio of data containing hate speech. The total amount of data after removing duplicates is 29.398 data. The example of crawled data can be seen in Table I.

TABLE I. EXAMPLE OF CRAWLED DATA

id_str	conversation_id_str	full_text
1640488563895566338	1640488563895566338	Hanya ngawasin Anies
1708981788171997200	1708981788171997200	Lalala anies ma cuman bisa ngedabrus, kayak lo yg pintar hoax makanya lulusan mantan napi kan ?

B. Data Labelling

The data obtained from crawling is assigned corresponding labels, where label 1 is given to data classified as hate speech, and label 0 is assigned to data that is not considered hate speech. With this approach, each piece of data has a binary classification indicating whether its content falls into the hate speech category (label 1) or not (label 0). This labeling process enables machine learning models to learn patterns and characteristics associated with hate speech for the purpose of classifying unseen data. The example of data labeling can be seen in Table II.

TABLE II. DATA LABELLING

Tweet	Label
“Iyain aja lah. Faktanya anies berhasil kolaborasi pangan saat masih menjabat gub dki”	0
“Berapa kali ya ngurus urusan administrasi kontol ini ga pernah bener anjing. Polisi, lurah, bpjs bisa ga buang2 waktu kah kontol. Cape banget ke sana ke mari tapi sama aja anjing.....”	1

C. Data Preprocessing

Several preprocessing methods are applied to the data as it is an essential process, so the data are easier to read by the model [24]. Removing duplicates from the data and cleaning the text data by removing symbols, characters, or emoticons that do not carry meaning is essential as they can disrupt the classification process. Case folding is also applied on the text data, converting all letter characters in the text to lowercase letter [25] to facilitate data classification process.

D. Data Splitting

The preprocessed data will then be divided into two parts, train and test data. The train data will be used to train the model for the classification task and the test data will be used to evaluate the performance of the model.

E. BERT Model Training

This stage begins with downloading a pre-trained BERT model. The pre-trained BERT model base that are used on this research is indoBERT, which is a BERT model that is pretrained on a large Indonesian dataset [15], as the base with an additional classification layer. This classification layer determines whether the input data are hate speech or non-hate speech. This classification layer will be fine-tuned to classify the data according to the labels.

To optimize the model's performance, Adam and AdamW Optimizers will be employed. During the training stage, the parameters of each layer in the BERT model will be adjusted to produce better predictions on the training data. After the model is trained with the training data and its parameters are optimized, the model will be used to predict the test data for evaluation.

The parameters that are going to be adjusted are specifically learning rate and weight decay. To optimize the learning rate, we will be using a learning rate scheduler. As shown in a study by Parmar et al (2020), the usage of learning rate scheduler for optimizer performs better than having a static learning rate [18]. As for weight decay fine-tuning, it is mentioned in a recent study by Loshchilov and Hutter [26] that the larger the batch pass is the smaller the optimal weight decay, so we took an approach to test the model with gradually smaller weight decay.

F. Evaluation

Evaluation is done by using the Confusion Matrix to calculate the metric of the performance of the model. Confusion Matrix is a method used to evaluate the

performance of a supervised algorithm by comparing the predicted results with the actual data. The Confusion Matrix can also be visualized in a table, as shown in Table III:

TABLE III. CONFUSION MATRIX

Class	Prediction	
	Positive (+)	Negative (−)
Positive (+)	TP	FN
Negative (−)	FP	TN

The True Positive (TP) value is obtained from the predicted positive classification that matches the actual data. Meanwhile, the False Positive (FP) value is the result of a positive classification prediction that is negative in the actual data. The same applies to True Negative (TN), where this value is obtained from the predicted negative classification that matches the actual data, and False Positive is the result of a negative classification prediction that has a positive class in the actual data.

The values obtained from the Confusion Matrix can then be used in the following equations to calculate the quality of the algorithm.

Accuracy is the comparison of the number of true positive and true negative to the overall classification results. Accuracy can be calculated using the following equation:

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

Precision is the comparison of the number of positively classified data that matches the actual data with the total number of positively classified data. Here is the equation to calculate precision:

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

Recall is the comparison of the number of positively classified data that matches the actual data with the total number of actual positive data, both those classified correctly and those that are not. Recall can be calculated using the following equation:

$$Recall = \frac{TP}{TP + FN} \quad (6)$$

IV. RESULT AND DISCUSSION

A. First Scenario

To achieve the best result on the training process, three different splitting ratios are tested on the baseline model. The result of the testing is as can be seen on Table IV:

TABLE IV. TRAINING DATA RATIO

Ratio	Accuracy (%)
90:10	49.49
80:20	49.64
70:30	**50.05**

It should be noted that there's no significant difference in the results. However, 70:30 ratio yields the best results out of the other two ratios with higher accuracy and lower loss. Therefore, 70:30 data splitting ratio will be applied for the next scenarios.

B. Second Scenario

The second scenario involves the application of AdamW and Adam Optimizer on the baseline model. Both optimizers are set to their default parameters in the pytorch library, which are 1e-3 for learning rate for both Adam and AdamW, and 1e-2 for AdamW's weight decay and 0 for Adam's weight decay.

TABLE V. OPTIMIZER

Optimizer	Accuracy (%)
AdamW	**50.05 (+0.00)**
Adam	49.64 (-0.41)

The results, as shown in Table V, show that the model with AdamW Optimizer performs better than model with Adam Optimizer in the default parameters with an accuracy of 50.05%. However as both results are barely above or below fifty percent, further investigation is obligatory for the research to achieve better performance on both models.

C. Third Scenario

In the third scenario we experimented by setting the initial learning rate of the optimizers into certain numbers to see which gives the best performance when applied. Learning rate values are chosen based on the graph that are obtained using the learning rate finder algorithm.

From Fig. 2 and Fig. 3, we estimated the recommended learning rate range for both Adam and AdamW are from 1e-4 to 1e-6. We use this range to test the impact of initial

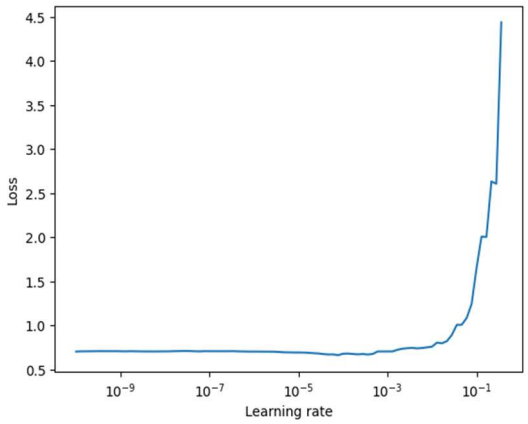

Fig. 2. Adam loss and learning rate graph.

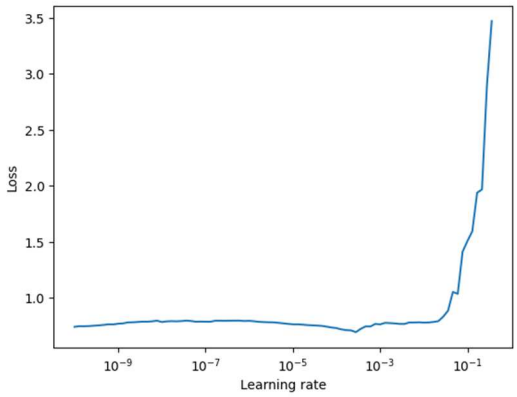

Fig. 3. AdamW loss and learning rate graph.

learning rates on the accuracy of the model, as can be seen in Table VI.

TABLE VI. INITIAL LEARNING RATE

Initial Learning Rate	Accuracy (%)	
	Adam	AdamW
1e-4	58.10 (+8.46)	50.02 (-0.03)
1e-4.5	89.93 (+40.29)	89.98 (+39.93)
1e-5	**90.03 (+40.38)**	**89.99 (+39.94)**
1e-6	89.27 (+39.63)	89.60 (+39.55)

There's a significant increase in accuracy when the learning rate value of 1e-4.5. This implies that the performance of the model heavily relies on the learning rate of the optimizer. With the learning rate of 1e-5, model with Adam Optimizer performs slightly better with the accuracy of 90.03% than the model with AdamW Optimizer with each using the default weight decay.

D. Fourth Scenario

In this scenario the value of the initial weight decay of both optimizers are set to a value starting from 0 which is the default value of Adam Optimizer's weight decay to AdamW Optimizer's default weight decay value, 1e-2. We take 1e-2 and 1e-3 as a comparison for both models.

TABLE VII. WEIGHT DECAY

Weight Decay	Accuracy (%)	
	Adam	AdamW
0	**90.03 (+40.38)**	90.03 (+39.98)
1e-1	86.91 (+37.27)	90.00 (+39.95)
1e-2	88.40 (+38.76)	89.99 (+39.94)
1e-3	89.87 (+40.23)	**90.08 (+40.03)**

The results, as can be seen in Table VII, show that model with AdamW optimizer performs best with 1e-3 as its initial weight decay with an accuracy of 90.08%. And the model with Adam performs best with its default initial weight decay with an accuracy of 90.03%. However, it should be noted that there is no significant gap between the accuracies.

E. Discussion

The results show that BERT model with AdamW performs best for hate speech classification task with as 1e-5 the learning rate and as 1e-3 the weight decay. With weight decay set to 1e-3, AdamW optimized BERT model with outperformed model with Adam Optimizer with an accuracy of 90.08%. Meanwhile the model with Adam reaches an accuracy of 90.03% with a learning rate of 1e-5 and with its default weight decay value. This shows that hyperparameters tuning for both Adam and AdamW improves model's performance for the hate speech classification task. There is a significant gap between the model before and after the learning rate is set from the default learning rate implying its importance in BERT model training. Weight decay also plays a role in increasing the accuracy of both models.

TABLE VIII. BEST MODELS PERFORMANCE

Performance	Adam	AdamW
F1-Score	95	94
Precision	96.6	95
Recall	93.2	90.8

Table VIII shows the performance of the best models for each label prediction evaluated with confusion matrix. While the model with AdamW has the best accuracy, model with Adam Optimizer has an overall better F1-Score, precision and recall value. This means that while the model with AdamW predicts more labels correctly, the model with Adam can identify data labelled as positive better.

V. CONCLUSION

In this research we built a model with a BERT based to be trained on hate speech detection with data crawled from Twitter in Bahasa. The models are optimized with Adam and AdamW. Then we fine-tuned the hyperparameters of the optimizers to achieve a better performance, such as learning rate and weight decay. This research found that fine-tuning the hyperparameters of the optimizer Adam and AdamW for hate speech classification task training on BERT model are proven to be able improve the accuracy of model significantly. By adjusting the learning rate parameter, the accuracy of the model increases by 39.94% from 50.05% in model with AdamW to 89.99% and from 49.64% in model with Adam to 90.03%, an increase of 40.38% from the baseline. Adjusting the weight decay also affects the accuracy of the model. By setting the accuracy to 1e-2 from the default 1e-2 in the model with AdamW, the accuracy improves to 90.08%, an increase of 40.03% from the baseline. Meanwhile the model with Adam has the best accuracy of 90.03% with its default weight decay parameter of 0. The model with AdamW has better accuracy than those of the model with Adam. However model with Adam has better overall f1-score, precision and recall than the other model. It is recommended for future works to fine-tune the hyperparameters of the optimizer to produce better results in the hate speech recognition task as it plays significant role that affects the model performance.

REFERENCES

[1] S. Kemp, "Digital 2023: Global overview report," 2023. [Online]. Available: https://datareportal.com/reports/digital-2023-global-overview-report. Accessed: May 8, 2023.

[2] S. Kemp, "Digital 2023: Indonesia," 2023. [Online]. Available: https://datareportal.com/reports/digital-2023-indonesia. Accessed: May 8, 2023.

[3] Twitter, "Twitter's policy on hateful conduct — Twitter's help," 2023. [Online]. Available: https://help.twitter.com/en/using-twitter/types-of-tweets. Accessed: May 15, 2023.

[4] A. Sepima, G. Siregar, and S. Siregar, "Penegakan hukum ujaran kebencian di Republik Indonesia," JURNAL RETENTUM, vol. 2, no. 1, pp. 108–116, 2021.

[5] D. I. Permatasari and S. Subyantoro, "Ujaran kebencian Facebook tahun 2017-2019," Jurnal Sastra Indonesia, vol. 9, no. 1, pp. 62–70, 2020.

[6] H. Saleh, A. Alhothali, and K. Moria, "Detection of hate speech using BERT and hate speech word embedding with deep model," Applied Artificial Intelligence, vol. 37, no. 1, pp. 2166719, 2023.

[7] S. Dowlagar and R. Mamidi, "Hasocone@fire-hasoc2020: Using BERT and multilingual BERT models for hate speech detection," 2021.

[8] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[9] T. Davidson et al., "Automated hate speech detection and the problem of offensive language," 2017.

[10] M. O. Ibrohim and I. Budi, "Multi-label hate speech and abusive language detection in Indonesian Twitter," in Proceedings of the Third Workshop on Abusive Language Online, Florence, Italy, Aug. 2019, pp. 46–57.

[11] D. J. Ningrum, S. Suryadi, and D. E. C. Wardhana, "Kajian ujaran kebencian di media sosial," Jurnal Ilmiah KORPUS, vol. 2, no. 3, pp. 241–252, Feb. 2019.

[12] J. Devlin, M.W. Chang, K. Lee, K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805v2*, 2019, https://doi.org/10.48550/arXiv.1810.04805.

[13] A. Ajagekar, "Adam," 2021. [Online]. Available: https://optimization.cbe.cornell.edu/index.php?title=Adam.

[14] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.

[15] F. Koto, A. Rahimi, J. H. Lau, and T. Baldwin, "IndoLEM and IndoBERT: A Benchmark Dataset and Pre-trained Language Model for Indonesian NLP," in Proceedings of the 28th COLING, 2020.

[16] V. W. Jonathan and E. B. Setiawan, "Ekspansi Fitur dengan GloVe untuk Deteksi Ujaran Kebencian Menggunakan Metode Convolutional Neural Network (CNN) dan Recurrent Neural Network (RNN) di Twitter," Jurnal Tugas Akhir Fakultas Informatika, 2023.

[17] M. F. Hidayat and E. B. Setiawan, "Ekspansi Fitur dengan GloVe untuk Deteksi Ujaran Kebencian Menggunakan Metode Convolutional Neural Network (CNN) dan Long Short-Term Memory (LSTM) di Twitter," Jurnal Tugas Akhir Fakultas Informatika, 2023.

[18] V. Parmar, N. Bhatia, S. Negi, and M. Suri, "Exploration of Optimized Semantic Segmentation Architectures for Edge-Deployment on Drones," arXiv:2007.02839 [cs.CV], 2020.

[19] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated Hate Speech Detection and the Problem of Offensive Language," arXiv:1703.04009 [cs.CL], 2017.

[20] M. O. Ibrohim and I. Budi, "Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter," in Proceedings of the Third Workshop on Abusive Language Online, Florence, Italy, August 2019, pp. 46-57, Association for Computational Linguistics.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," arXiv:1706.03762 [cs.CL], 2023.

[22] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to Fine-Tune BERT for Text Classification?," arXiv:1905.05583 [cs.CL], 2020.Afa

[23] M. McCloskey and N. J. Cohen, "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem," in Psychology of Learning and Motivation, vol. 24, pp. 109–165, Elsevier, 1989.

[24] A. Krouska, C. Troussas, and M. Virvou, "The effect of preprocessing techniques on Twitter sentiment analysis," in Proceedings of the 7th International Conference on Information, Intelligence, Systems and Applications (IISA), July 2016, pp. 1-5, doi: 10.1109/IISA.2016.7785373.

[25] M. Rosid, A. Fitrani, I. Astutik, N. Mulloh, and H. Gozali, "Improving Text Preprocessing for Student Complaint Document Classification Using Sastrawi," IOP Conference Series: Materials Science and Engineering, vol. 874, pp. 012017, July 2020, doi: 10.1088/1757-899X/874/1/012017.

[26] I. Loshchilov and F. Hutter, "Fixing Weight Decay Regularization in Adam," 2018. [Online]. Available: https://openreview.net/forum?id=rk6qdGgCZ.