IndoBERT Based Data Augmentation for Indonesian Text Classification

Fuad Muftie
Computer Science Study Program
Universitas Nusa Mandiri
Jakarta, Indonesia
14210197@nusamandiri.ac.id

Muhammad Haris
Computer Science Study Program
Universitas Nusa Mandiri
Jakarta, Indonesia
muhammad.uhs@nusamandiri.ac.id

***Abstract*—Text data augmentation has been able to improve the performance of models or algorithms for text classification and sentiment analysis. In major cases, rule-based augmentation techniques can be easily applied to various languages such as Indonesia language. However, language model-based augmentation techniques rarely investigated for Indonesian language data. Therefore, this paper presents a text data processing model for Indonesian language, which has limited data, to perform text preprocessing and data augmentation techniques by selectively inserting words based on IndoBERT. This IndoBERT-based augmentation is able to generate data that still retains meaning and sentiment similar to the original data. The testing of this Twitter text dataset yielded results showing that the proposed augmentation technique was able to increase accuracy and outperform the Random Insert augmentation technique.**

Keywords—data augmentation, indobert, text preprocessing, cnn, bi-lstm

I. INTRODUCTION

Natural Language Processing (NLP) remains a rapidly evolving research topic alongside the advancement of Deep Learning [1]. NLP encompasses a wide range of fields, including text classification, document summarization, text translation, sentiment analysis, chatbots, text generation, speech recognition, and more. With the continuous development of Deep Learning technology, there are also increasing discoveries in this field. However, one of the challenges in NLP is the availability of large-scale data. While textual data is abundant, there are still limitations in terms of specific purposes. Deep Learning for NLP requires large amounts of data to obtain strong models. Limited data can lead to model overfitting [2].

To overcome the limitations of data availability, data augmentation techniques are employed as a strategy to increase the training data without the need for collecting new data [3]. Data augmentation for NLP has been receiving attention from researchers, and the need for data augmentation in various industries is also increasing. Text data augmentation still faces significant challenges and lags behind image data augmentation methods, as even slight changes in text can alter the meaning of a sentence [1]. Existing techniques for text data augmentation, such as deletion, insertion, or substitution of words and letters within a sentence, often result in ambiguous sentence meanings and altered sentiment [4]. Thus, an ideal data augmentation technique for text would involve paraphrasing [5].

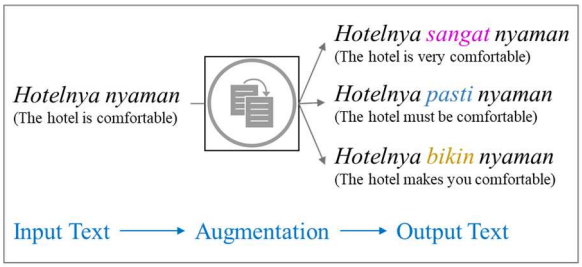

Fig. 1. Text data augmentation model for Indonesian language

Data augmentation is a method to add the diversity data training without collecting new data. Typically, data augmentation involves making slight modifications to the data data or generating synthetic data [6][7]. Processing techniques such as cropping, flipping, and jittering are commonly used. However, in the field of NLP, effectively generating synthetic data through augmentation has not yet achieved uniform implementation [3].

Despite the challenges associated with text data, several data augmentation techniques such as text manipulation [5] and generative approaches [8] have been implemented. Most techniques offer a trade-off between ease of implementation and performance. Rule-based techniques are easy to implement but only slight improvements are achieved [9][10]. Meanwhile, generative models usually requires more computations but offer better performance in general. Customizing pre-trained model-based techniques for specific domains can have a significant impact on model performance but can be challenging to develop and apply to other fields.

Furthermore, the distribution of augmented data should not be too similar or too different from the training data. This can lead to overfitting and hence the performance would suffer. Data augmentation should be able to generate generalizations so that the model performs well on seen and unseen data. Generalization remains a significant challenge has led to the proposal of data augmentation to achieve better generalization [11].

Text data augmentation has been a prominent focus of research in NLP development in currently, where limited training data is automatically transformed into a larger

2023 International Conference on Information Technology Research and Innovation (ICITRI) | 979-8-3503-2494-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICITRI59340.2023.10250061

quantity of data without the need for collecting new data. In [12], back translation technique is used to generate new data. Similarly, Shleifer [13] employed back translation to increase the amount of data, by translating English texts (IMDB dataset) into a second language and then translating them back into English. This method significantly improved the model's accuracy, reaching up to 80.6%.

Wei and Zou [9] applied several augmentation techniques called Easy Data Augmentation (EDA), which involve creating augmentation rules using the synonyms instead and randomly swapping words in a sentence, randomly inserting specific words into sentences, and randomly deleting some words to generate new data. Although these techniques simple and do not rely on any language models, they substantially improve model performance.

In [4], semi-supervised learning approach is used to perform sentiment analysis on four Vietnamese text datasets with limited data. These Vietnamese datasets required various data preprocessing techniques to clean and normalize the data, handle negations, and handle intensifiers to improve model performance. The data augmentation techniques used included back translation, syntax-tree transformation, and EDA. Meanwhile in [1] A data augmentation method called Hierarchical Data Augmentation (HDA) is applied to text classification. Inspired by the hierarchical structure of texts, where words form sentences and sentences form documents, HDA applies hierarchical data augmentation strategies by adding text at both the word and sentence levels. This method was also inspired by the cropping method used for image data augmentation. At each augmentation level, HDA hierarchically filters (cuts) content from the text, extracting the most important parts of the sentences to create text summaries. Data augmentation using HDA can generate a large amount of high-quality text data.

These various studies have demonstrated that text augmentation techniques, particularly in the English language, have significantly improved machine learning model performance. Therefore, this paper conducts experiments to enhance the accuracy of models for Indonesian language text classification with datasets that have limited data in their training set. The techniques used include text preprocessing to clean and normalize the data, as well as automatically increasing the number of training samples using BERT-based data augmentation techniques. To evaluate this approach, several classification algorithms for well-known languages such as CNN and Bi-LSTM will be applied.

The main contribution of this research lies in the selection of text preprocessing methods for *Bahasa Indonesia* and the use of BERT in the Indonesian language to create synthetic data from limited text data by selectively inserting words into sentences. Illustration examples of text data augmentation models in Indonesian language can be seen in Fig. 1, where the input text undergoes augmentation using BERT-based methods (IndoBERT), resulting in new data that still maintains the same meaning and sentiment as the original data.

II. METHOD

The research methodology in this paper is structured as shown in Fig. 2.

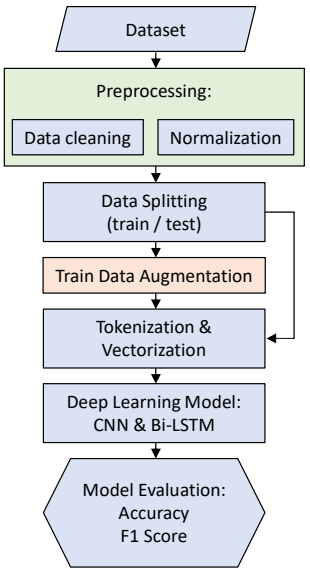

Fig. 2. Research Methods Chart

A. Dataset

Two datasets were used in this study. The first dataset is "The Dataset for Hate Speech Detection in Indonesian" [14], which contains 713 rows of Indonesian-language Twitter text. The acquired data consists of a predictor variable, which is the tweet text, and a target variable with two classes (Hate Speech / HS and Non_HS). The distribution of the two classes is as follows: 453 rows of Non_HS data (63.53%) and 260 rows of HS data (36.47%).

The second dataset is the "Twitter Sentiment Analysis Dataset," which consists of 782 rows of Twitter text. This dataset has two classes: positive sentiment (398 data/50.90%) and negative sentiment (384 data/49.10%).

B. Data Processing

In this stage, preprocessing is applied to both the predictor and target variables. The tweet text (predictor variable) undergoes data cleaning, which includes removing unnecessary words, emojis, non-ASCII special characters, URLs, HTML elements, standalone numbers, and converting the text to lowercase (case folding). After cleaning the text data, normalization is performed to convert informal words and abbreviations into more formal and consistent words. The data is then split into training and test sets. Data augmentation is applied to the training data to increase the number of training samples using two augmentation techniques as a comparison, first using Random Insert as part of the EDA and second using a language model-based augmentation technique with IndoBERT.

Next, the cleaned, normalized, and augmented text data undergo feature engineering, including tokenization and

vectorization, to transform the text data into vectorized form suitable for Deep Learning models.

C. Model Development

Deep Learning (DL) modeling is employed, which is a neural network-based technique widely used in recent years as a method of implementing Machine Learning (ML). DL refers to the learning process within Deep Neural Networks (DNNs). DNNs are neural networks with multiple layers (input layer, N hidden layers, output layer), typically consisting of more than three layers [15]. The DL algorithms used in this study are CNN and Bi-LSTM. The entire models are implemented in a Jupyter Notebook environment using Python, Scikit-Learn, and TensorFlow/Keras. The models are tested with both augmented and non-augmented data to compare their performance.

CNN belongs to the category of Deep Learning algorithms, which is an extension of multilayer perceptron (MLP) designed to process data in grid form, including two-dimensional images. CNN is a type of feedforward neural network known for its excellent performance in image processing and natural language processing [16]. The main characteristic of CNN is its ability to recognize and extract various prominent features from a given perspective, making it commonly used for feature engineering, particularly in feature extraction. One of the key advantages of CNN is its local perception and weight sharing, which significantly reduces the number of parameters and improves learning efficiency. CNN mainly consists of two parts: convolutional layers and pooling layers. Each convolutional layer contains a number of convolutional kernels.

LSTM is a variant of the Recurrent Neural Network (RNN) unit. LSTM considers the possible presence of unknown time gaps between important events [17]. LSTM has a limitation in that it can only process words in a sentence in one direction. Bidirectional Long Short-Term Memory (BiLSTM) is an enhanced version of LSTM that overcomes this limitation by processing words in both directions [18][19].

D. Evaluation

The evaluation process is a stage to test the accuracy of the model by making predictions using predetermined test data. This study uses accuracy and F1 Score as metrics to measure and evaluate the model. Accuracy measures all correctly predicted results compared to the total predictions made. Accuracy metrics are commonly used for datasets where all labels are considered equally important, including both label 0 and label 1. Accuracy is also used when True Positive (TP) and True Negative (TN) predictions are considered more important than others. Accuracy is defined as in formula 1.

$$accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \quad (1)$$

On the other hand, F1 Score is the harmonic mean of precision and recall, calculated by:

$$F1\ Score = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \quad (2)$$

where TP = True Positive, TN = True Negative, FP = False Positive and FN = False Negative.

Additionally, an evaluation of the augmentation results is performed using the BLEU (Bilingual Evaluation Understudy) score to compare the augmented data/sentences against the original data. The BLEU score indicates the accuracy (goodness) of the imitation sentences compared to the original sentences.

III. Results and Discussion

The experiments were conducted in the Jupyter Notebook environment using Python version 3.7. First, the dataset was cleaned. The data cleaning process followed the following order: removing emojis and their Unicode representations, removing URLs and HTML elements like HTML tags, removing non-ASCII characters, removing newline characters, removing email addresses, removing Twitter mentions, removing punctuation marks, removing standalone numbers, removing standalone single letters, and converting multiple spaces into single spaces.

Text normalization was performed using an internet slang dictionary in Indonesian language that contains a list of commonly used informal words and abbreviations on Twitter, along with their formal counterparts. Through normalization, the text is transformed into a formal and consistent form, ensuring the same representation for words with the same meaning.

After the data cleaning and normalization process, duplicate data, empty data, and single-word data were checked and removed. Next, the data was split into training data (80%) and test data (20%).

The training data was then augmented in two ways. The first method involved randomly inserting a word using the Random Insert (EDA) package by Wei and Zou [9]. The augmented data obtained from this method was used as a benchmark for the proposed augmentation method. The second method utilized an augmentation technique based on IndoBERT, a language model, to select relevant words that maintain the original sentence's meaning and sentiment. The results of both augmentation methods were compared with the original data without augmentation.

TABLE I. Example Of Random Insert Augmentation

Original sentence	Hotel tempat kami menginap sangat nyaman (*The hotel where we stayed was very comfortable*)
Random Insert	Hotel tempat kami resort menginap sangat nyaman *(The hotel where we resort stayed was very comfortable)*

This paper uses IndoBERT package from Indolem called IndoBERTweet, which is the Indonesian version of BERT, as the first large-scale pre-trained model for Indonesian Twitter. It expands the Indonesian BERT model trained monolingually with domain-specific additive vocabulary [20].

TABLE II. EXAMPLE OF PROPOSED METHOD

Original sentence	Hotel tempat kami menginap sangat nyaman (*The hotel where we stayed was very comfortable*)
POS Tagging	Hotel tempat kami menginap sangat nyaman NN NN PRP VB RB JJ
Insert MASK	Hotel tempat kami menginap [MASK] sangat nyaman
Insert word using IndoBERT	Hotel tempat kami menginap bersama sangat nyaman *(The hotel where we stayed together was very comfortable)*

The process of text data augmentation begins with identifying the POS tagging of each sentence to find the positions of VB (verb), IN (preposition), JJ (adjective), CC (coordinating conjunction), and SC (subordinating conjunction). If there is only one VB, insertion is done after the VB. If there are multiple VBs, insertion is prioritized after the second VB. If no VB is found, IN is searched for insertion. If no VB or IN is found, JJ is searched for insertion. If no VB, IN, or JJ is found, CC is searched for insertion. If no VB, IN, JJ, or CC is found, SC is searched for insertion. If none of these tags (VB, IN, JJ, CC, and SC) are found, random word insertion is performed.

The word insertion technique for these tags uses the [MASK] token at the targeted position within the sentence or original data. The sentence with the masked token is then unmasked using IndoBERT to obtain 10 candidate words to be selected for insertion. From these ten candidate words, filtering is applied to meet the criteria: not punctuation, not the same as the preceding or succeeding word, not a negation word, not a conjunction word, not a numeric value, and not a subword. If there are words that meet these criteria among the 10 insertion candidate words, the first word is selected for insertion as an unmasked word. This proposed text data augmentation process ensures coherent sentences with preserved meaning and sentiment. This process ensures good generalization compared to random insertion techniques that may render the sentence meaningless or alter its sentiment.

The original training data, training data augmented with random insertion, training data augmented with the proposed technique, and testing data are all tokenized and vectorized to transform the text data into vector data for use in deep learning models.

The training and testing process on the CNN and Bi-LSTM models is performed using each of the following: the original training data before augmentation, training data after augmentation with random insertion, and training data after augmentation with the proposed technique. Each test is conducted ten times, and the average results are recorded. The testing results, along with the testing data, are presented in Tables III to VI.

TABLE III. ACCURACY SCORE WITH CNN

Dataset	Base	Random Insert	IndoBert
Dataset 1	0.8681	0.8773	**0.8801**
Dataset 2	0.6369	0.6554	**0.6662**

TABLE IV. F1 SCORE WITH CNN

Dataset	Base	Random Insert	IndoBert
Dataset 1	0.8151	0.8233	**0.8281**
Dataset 2	0.6658	**0.6899**	0.6845

TABLE V. ACCURACY SCORE WITH BI-LSTM

Dataset	Base	Random Insert	IndoBert
Dataset 1	0.8369	0.8624	**0.8667**
Dataset 2	0.6153	0.6471	**0.6516**

TABLE VI. F1 SCORE WITH BI-LSTM

Dataset	Base	Random Insert	IndoBert
Dataset 1	0.7958	**0.8174**	0.8142
Dataset 2	0.6408	0.6661	**0.6921**

In the first test using the CNN model, the accuracy results are as presented in Table III, and the F1 scores are shown in Table IV. First, using the original data (before augmentation) as the baseline, the accuracy achieved is 0.8681 (dataset 1) and 0.6369 (dataset 2). Second, using data augmented with random insertion, the accuracy achieved is 0.8773 (dataset 1) and 0.6554 (dataset 2). This indicates an improvement in accuracy compared to the baseline. Third, using data augmented with the proposed technique, the accuracy achieved is 0.8801 (dataset 1) and 0.6662 (dataset 2). Thus, the proposed augmentation technique is capable of increasing accuracy compared to random insertion.

Meanwhile, the F1 score in the CNN model shows that, first, using the original data (before augmentation) as the baseline, an F1 score of 0.8151 (dataset 1) and 0.6658 (dataset 2) is obtained. Second, using data augmented with random insertion, an F1 score of 0.8233 (dataset 1) and 0.6899 (dataset 2) is obtained. Third, using data augmented with the proposed augmentation technique, an F1 score of 0.8281 (dataset 1) and 0.6899 (dataset 2) is obtained. Thus, the Random Insert augmentation technique is capable of increasing the F1 score compared to the baseline. However, the proposed augmentation technique is able to increase the F1 score compared to Random Insert in dataset 1, while in dataset 2, the Random method is still superior to other methods.

The second test uses the Bi-LSTM model, with accuracy results as shown in Table V and F1 scores in Table VI. First, using the original data (before augmentation) as the baseline, an accuracy of 0.8369 (dataset 1) and 0.6153 (dataset 2) is achieved. Second, using data augmented with random insertion, an accuracy of 0.8624 (dataset 1) and 0.6471 (dataset 2) is achieved. This also indicates an improvement in accuracy compared to the baseline. Third, using data augmented with the proposed augmentation technique, an accuracy of 0.8667 (dataset 1) and 0.6516 (dataset 2) is achieved. Thus, the proposed augmentation technique is capable of increasing accuracy and outperforming the Random Insert augmentation technique.

Regarding the F1 scores in the Bi-LSTM model, first, using the original data (before augmentation) as the baseline,

an F1 score of 0.7958 (dataset 1) and 0.6408 (dataset 2) is obtained. Second, using data augmented with random insertion, an F1 score of 0.8174 (dataset 1) and 0.6661 (dataset 2) is obtained. Third, using data augmented with the proposed augmentation technique, an F1 score of 0.8142 (dataset 1) and 0.6921 (dataset 2) is obtained. Thus, the Random Insert augmentation technique is capable of increasing the F1 score compared to the baseline. However, the proposed augmentation technique is able to increase the F1 score compared to Random Insert in dataset 2, while in dataset 1, the Random method is still superior to other methods.

Lastly, the BLEU score is calculated individually for the IndoBERT-based augmented data compared to each respective original data. The overall test results are averaged, resulting in an average total score of 88.91% for dataset 1 and 88.25% for dataset 2.

IV. CONCLUSION

The selective insertion augmentation method, as proposed, is capable of increasing the accuracy score compared to the Random Insert method in both dataset 1 and dataset 2, using both the CNN and Bi-LSTM models. Meanwhile, the F1 score for the proposed augmentation method outperforms the Random Insert method in Dataset 1 with the CNN model and in Dataset 2 with the Bi-LSTM model. Therefore, overall, the selective insertion method using IndoBERTweet can assist in improving the performance of Deep Learning models, specifically the CNN and Bi-LSTM models, as demonstrated in the presented tests. For future development of the selective insertion method, experiments can be conducted by inserting more words into sentences, especially for long text data that require more data for use in Deep Learning.

ACKNOWLEDGMENT

The authors would like to thank all those who have worked with us so far and other related projects.

REFERENCES

[1] S. Yu, J. Yang, D. Liu, R. Li, Y. Zhang, and S. Zhao, “Hierarchical Data Augmentation and the Application in Text Classification,” *IEEE Access*, vol. 7, pp. 185476–185485, 2019, doi: 10.1109/ACCESS.2019.2960263.

[2] M. Bayer, M.-A. Kaufhold, B. Buchhold, M. Keller, J. Dallmeyer, and C. Reuter, “Data augmentation in natural language processing: a novel text generation approach for long and short text classifiers,” *International Journal of Machine Learning and Cybernetics*, vol. 14, pp. 135-150, Apr. 2022, doi: 10.1007/s13042-022-01553-3.

[3] S. Y. Feng, V. Gangal, J. Wei, S. Chandar, S. Vosoughi, T. Mitamura, et al., "A survey of data augmentation approaches for NLP", Proc. Findings of the Association for Computational Linguistics, pp. 968-988, 2021.

[4] H.-T. Duong and T.-A. Nguyen-Thi, “A review: preprocessing techniques and data augmentation for sentiment analysis,” *Comput Soc Netw*, vol. 8, no. 1, p. 1, 2021, doi: 10.1186/s40649-020-00080-x.

[5] X. Zhang and Y. LeCun, “Text Understanding from Scratch,” arXiv:1502.01710, 2015.

[6] C. Shorten and T. M. Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning,” *J Big Data*, vol. 6, no. 1, p. 60, 2019, doi: 10.1186/s40537-019-0197-0.

[7] A. Hernández-García and P. König, “Data augmentation instead of explicit regularization,” arXiv:1806.03852, 2020.

[8] R. Liu, G. Xu, C. Jia, W. Ma, L. Wang, and S. Vosoughi, “Data Boost: Text Data Augmentation Through Reinforcement Learning Guided Conditional Generation,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online: Association for Computational Linguistics, Nov. 2020, pp. 9031–9041. doi: 10.18653/v1/2020.emnlp-main.726.

[9] J. Wei and K. Zou, “EDA: Easy data augmentation techniques for boosting performance on text classification tasks,” *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, vol. abs/1901.1, pp. 6382–6388, 2019, doi: 10.18653/v1/d19-1670.

[10] J. Wei, C. Huang, S. Xu, and S. Vosoughi, “Text Augmentation in a Multi-Task View,” in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Online: Association for Computational Linguistics, Apr. 2021, pp. 2888–2894. doi: 10.18653/v1/2021.eacl-main.252.

[11] R. Raileanu, M. Goldstein, D. Yarats, I. Kostrikov, and R. Fergus, “Automatic Data Augmentation for Generalization in Reinforcement Learning,” in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. W. Vaughan, Eds., Curran Associates, Inc., 2021, pp. 5402–5415. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/2b38c2df6a49b97f706ec9148ce48d86-Paper.pdf

[12] A. Sugiyama and N. Yoshinaga, “Data augmentation using back-translation for context-aware neural machine translation,” in *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 35–44. doi: 10.18653/v1/D19-6504.

[13] S. Shleifer, “Low resource text classification with ulmfit and backtranslation,” *arXiv preprint arXiv:1903.09244*, 2019.

[14] I. Alfina, R. Mulia, M. I. Fanany, and Y. Ekanata, “Hate speech detection in the Indonesian language: A dataset and preliminary study,” in *2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, 2017, pp. 233–238. doi: 10.1109/ICACSIS.2017.8355039.

[15] C. I. Agustyaningrum, M. Haris, R. Aryanti, T. Misriati, and others, “Online shopper intention analysis using conventional machine learning and deep neural network classification algorithm,” *Jurnal Penelitian Pos dan Informatika*, vol. 11, no. 1, pp. 89–100, 2021.

[16] B. S. Kim and T. Kim, “Cooperation of Simulation and Data Model for Performance Analysis of Complex Systems,” *International Journal of Simulation Modelling*, vol. 18, pp. 608–619, 2019, doi: 10.2507/IJSIMM18(4)491.

[17] A. Yadav, C. K. Jha, and A. Sharan, “Optimizing LSTM for time series prediction in Indian stock market,” *Procedia Comput Sci*, vol. 167, pp. 2091–2100, 2020, doi: https://doi.org/10.1016/j.procs.2020.03.257.

[18] H. Elfaik and E. H. Nfaoui, “Deep Bidirectional LSTM Network Learning-Based Sentiment Analysis for Arabic Text,” *Journal of Intelligent Systems*, vol. 30, no. 1, pp. 395–412, 2021, doi: doi:10.1515/jisys-2020-0021.

[19] S. Ahmed, A. F. Saif, M.S. Hanif, M. M. N, Shakil, M.M. Jaman, et al., "Att-BiL-SL: Attention-Based Bi-LSTM and Sequential LSTM for Describing Video in the Textual Formation", Appl. Sci, vol. 12, pp. 317, 2022.

[20] F. Koto, J. H. Lau, and T. Baldwin, “IndoBERTweet: A Pretrained Language Model for Indonesian Twitter with Effective Domain-Specific Vocabulary Initialization,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural LanguageProcessing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 10660–10668. doi: 10.18653/v1/2021.emnlp-main.833.