On the Role of Text Preprocessing in BERT Embedding-based DNNs for Classifying Informal Texts

Aliyah Kurniasih
Department of Computer Science
University of Nusa Mandiri
Jakarta, Indonesia

Lindung Parningotan Manik
Research Center for Data and Information Sciences
National Research and Innovation Agency
Cibinong, Indonesia

***Abstract*—Due to highly unstructured and noisy data, analyzing society reports in written texts is very challenging. Classifying informal text data is still considered a difficult task in natural language processing since the texts could contain abbreviated words, repeating characters, typos, slang, et cetera. Therefore, text preprocessing is commonly performed to remove the noises and make the texts more structured. However, we argued that most tasks of preprocessing are no longer required if suitable word embeddings approach and deep neural network (DNN) architecture are correctly chosen. This study investigated the effects of text preprocessing in fine-tuning a pre-trained Bidirectional Encoder Representations from Transformers (BERT) model using various DNN architectures such as multilayer perceptron (MLP), long short-term memory (LSTM), bidirectional long-short term memory (Bi-LSTM), convolutional neural network (CNN), and gated recurrent unit (GRU). Various experiments were conducted using numerous learning rates and batch sizes. As a result, text preprocessing had insignificant effects on most models such as LSTM, Bi-LSTM, and CNN. Moreover, the combination of BERT embeddings and CNN produced the best classification performance.**

Keywords*—Natural language processing; bert embeddings; deep neural network; text preprocessing

I. Introduction

Societies often generate informal texts in the form of complaints, aspirations, and ideas. Therefore, it is crucial to follow up on most of the reports received by the various applications to increase service quality. However, the fluid, social, and dynamic written language that continues to develop becomes a challenge in natural language processing (NLP) research field. Furthermore, other challenges in text data such as typos, slang words, and informal vocabularies, followed by various hashtags and emoticons, remain to continue.

In order to overcome these problems, text preprocessing is performed to manage text data before building an NLP model using machine learning. For example, removing hashtags, URLs, stopwords, punctuations, @annotation, ASCII, and duplicate characters in a word is common in text mining [1]. Furthermore, tokenization, case-folding, stemming, and lemmatization were also performed preprocessing texts [2]. These steps are important in conventional machine learning since preprocessing can decrease vocabulary size by removing unhelpful parts of data or noise [3]. Thus, it can reduce the text data size and enhance the effectiveness and efficiency of the machine learning algorithms.

However, these approaches could be problematic. Some preprocessing steps could make semantic meaning between tokens or words in sentences disappear. For example, removing some stopwords could affect the contexts and generate ambiguous results. Sometimes, emoticons and hashtags could be helpful when analyzing emotions or sentiments within texts. Moreover, mistakes could be made if done manually or even automatically. To the best of our knowledge, no stemmer has 100% accuracy. Thus, in addition to losing the meaning, over stemming and under stemming could occur.

There are several techniques for extracting text features. Within text mining, feature extraction means converting texts to vectors. In conventional machine learning, the Bag-of-Word (BOW) method and Term Frequency-Inverse Document Frequency (TF-IDF) are commonly used [4]. The deficiency of these approaches is that they do not capture the position in the text, co-occurrences in different documents, and the semantics. Some of these problems were solved with word embeddings, which are learned text representations in which words with related meanings are represented similarly. It is considered one of the breakthroughs in deep learning. Studies in [5], [6], [7] suggested the Word2Vec approach to extract text features, while others suggested Glove [8]. Nevertheless, both approaches are context-independent, and they could not catch all semantic information such as Out-Of-Vocabulary (OOV) and some opposite word pairs.

Now-a-days, the NLP model that could perfectly capture almost all semantic contextual meanings is the Bidirectional Encoder from Transformers (BERT) [9]. It takes a sequence (typically a sentence) as input rather than a single word to generate contextual embeddings. Before BERT can build word embeddings, the context provided by surrounding words has to be shown. Word2Vec generates only one vector representation for each word. If there are any different word meanings, they are combined into one single vector. Meanwhile, BERT generates different vectors of a single word in different contexts. It is a leap in text mining techniques where pre-trained models are utilized in transfer learning with Transformers network [10]. Some pre-trained BERT models are already available in some languages other than English, such as AnchiBERT for ancient Chinese language [11], PhoBERT for Vietnamese [12], and

IndoBERT for the Indonesian language [13], [14].

In previous studies, BERT has been applied in text classification and generated passably result [15], [16], [17], [18], [19]. With many noises in the data, such as slang words, non-standard abbreviations, and typos, experiments conducted in [20] to analyze the sentiments of flood disaster-related texts using a pre-trained BERT model showed promising results. The study claimed that the noises had great effects on accuracy. However, the authors did not experiment with text preprocessing to prove that claim. We hypothesized that choosing the suitable word embeddings approach and DNN architecture makes most text preprocessing steps no longer required.

The contribution of this study is twofold. First, we investigated the effect of text preprocessing in the BERT embedding-based deep neural networks (DNNs) when classifying informal texts. While NLP studies suggested that text preprocessing was required most of the time, we argued that these tasks do not affect classification performance nowadays. Second, we also aimed to find and propose DNN architecture with the best classification performance in fine-tuning BERT embeddings. Therefore, we conducted experiments with or without most text preprocessing tasks using five DNN architectures, including long short-term memory (LSTM), bidirectional long short-term memory (Bi-LSTM), convolutional neural network (CNN), multi-layer perceptron (MLP), and gated recurrent unit (GRU). In addition, each model was also tuned with various optimization methods and hyperparameters, such as learning rate and batch size.

The remainder of this paper is organized as follows. First, Section II presents the dataset and research methods. Then, the results and discussions are explained in Section III and Section IV, respectively. Finally, conclusions and future research recommendations are provided in Section V.

II. Materials and Methods

The research method in this study is shown in Fig. 1. First, we collected a dataset in the Indonesian language from society reports taken from a Citizen Relation Management (CRM) application in the Water Resources Agency of Jakarta, Indonesia. There were 3,217 instances obtained from 1 January to 31 July 2021. Initially, the CRM administration staff performed the manual classifications of the text reports into five handling categories, namely flood mitigation, waterways, drain closure, infiltration well, and others. The distribution of data is displayed in Table I.

TABLE I. Number of Labeled Data

Category	Amount of Data
Flood mitigation	1360
Waterways	1071
Others	423
Drain closure	343
Infiltration well	20

We performed text preprocessing semi-automatically before conducting one of our sequences of experiments. Correcting abbreviated vocabulary, removing repeated syllable alphabet, repairing typos, and formalizing slang words to standard words based on Indonesian dictionary rules were done manually. Meanwhile, case-folding and removing numbers, mentions, hashtags, as well as emoticons were performed automatically.

The IndoBERT model released by IndoNLU [13] was used to create the BERT embeddings of the dataset. The model was pre-trained using Masked Language Modelling (MLM) and Next Sentence Prediction (NSP), consisting of 124.5M parameters in the base architecture. A text data collection which consists of 4 billion words called Indo4B with a size of 23.43 GB, was used to train the model.

Furthermore, we divided the dataset into 70% training data, 15% validation data, and 15% test data. Training data used to build model, validation data used to test the trained networks to validate model, and testing data used to test model that was built. Validation and test data used were not part of training data to produce an objective evaluation result. Moreover, five DNN architectures were trained. The LSTM, Bi-LSTM, CNN, MLP, and GRU were chosen since they performed well in previous studies [1], [8], [21].

TABLE II. Hyperparameters of DNN Algorithms

Parameter	Parameter Value
IndoBERT	Indobenchmark/indobert-base-p1
Max length	512
Neuron	1024, 512, 256
Batch size	16, 32
Dropout	0.2
Activation function	ReLu
Output function	Softmax
Loss function	Categorical_crossentropy
Epoch	20
Number of layer	1-5
Type of layer	LSTM, BiLSTM, CNN, GRU, MLP
Optimization	Adam
Learning rate	$5x10^{-5}$, $3x10^{-5}$, $2x10^{-5}$

TABLE III. Models' Architecture

Architecture	Layer
MLP	One input layer, three dense layer (neuron 1024, 512, 256), dropout layer (0.2), one output layer
LSTM	One input layer, one lstm layer (1024), two dense layer (512, 256), dropout layer (0.2), one output layer
BiLSTM	One input layer, one bi-lstm layer (1024), two dense layer (512, 256), dropout layer (0.2), one output layer
CNN	One input layer, one convolutional layer (1024), one pooling layer, two dense layer (512, 256), dropout layer (0.2), one output layer
GRU	One input layer, one gru layer (1024), two dense layer (512, 256), dropout layer (0.2), one output layer

Hyperparameter values such as maximum length, the number of neurons, learning rates, batch sizes, and epochs were determined based on previously conducted research related to BERT fine-tuning [15]. Moreover, the ReLu activation function was used on the hidden layer and the Softmax activation function on the output layer. The categorical cross-entropy was used as a loss function since target label classification has more than two classes. The dropout value was set to 0.2 used on the last hidden layer before the output layer to regularize the model to decrease overfitting from happening on the model.

Each model was built and experimented with a variation

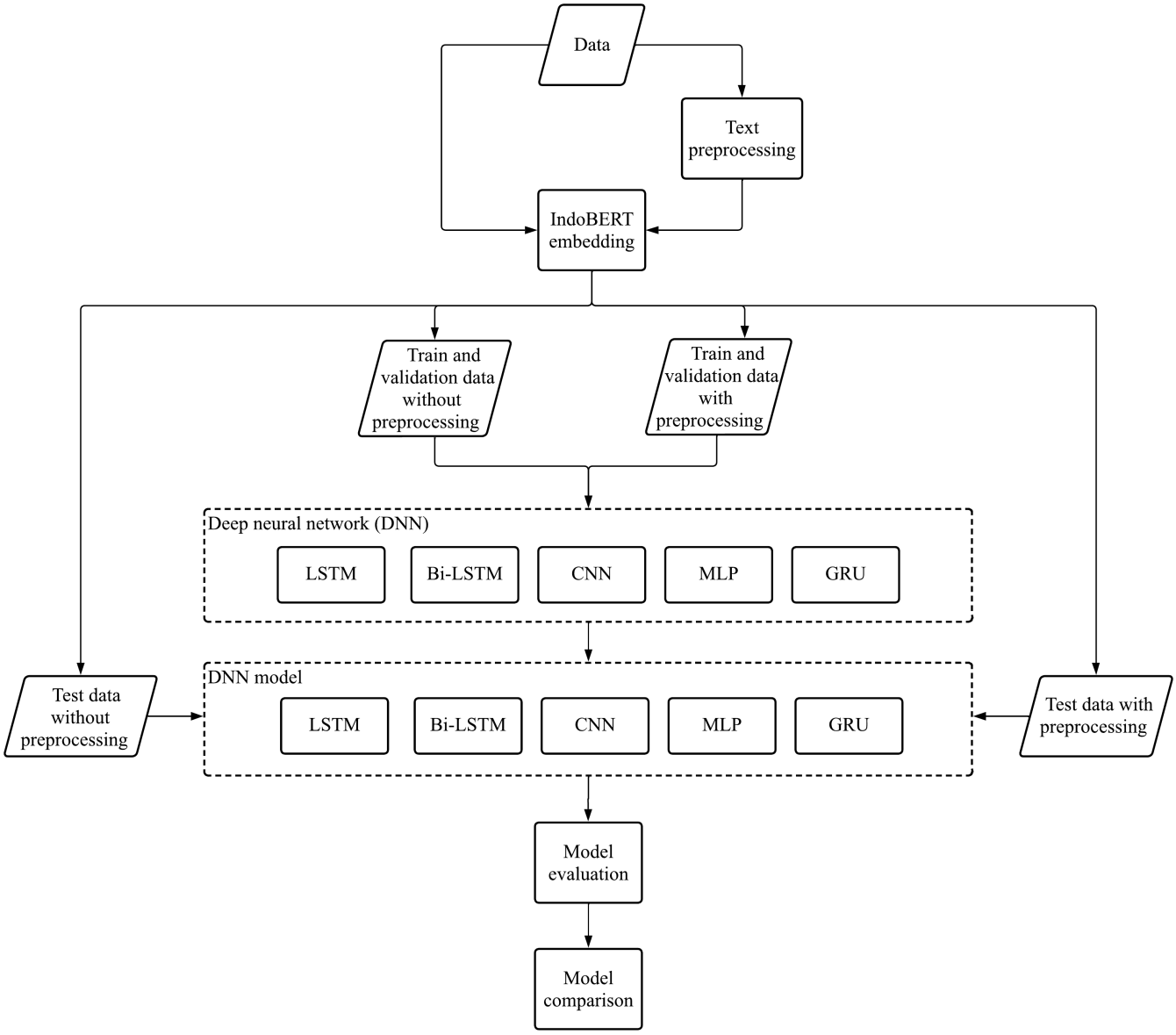

Fig. 1. Research Method.

amount of batch size of 16 as well as 32, learning rates of $5x10^{-5}$, $3x10^{-5}$, $2x10^{-5}$, and an epoch of 20. Moreover, Adam is used to optimizing the model since it is the best adaptive optimizer in most cases. The summary of the hyperparameter values is shown in Table II. Meanwhile, all DNN architectures were composed of one input layer, one output layer, and at least one hidden layer. The detail of the five architectures and the used parameters on the layers is represented in Table III.

Standard performance metrics, such as TP (True Positive), TN (True Negative), FP (False Positive), and FN (False Negative), were used as primary building blocks to evaluate classification models. A TP is measured when the model predicts the positive class correctly. A TN, on the other hand, is a result in which the model correctly classifies the negative class. Conversely, a FP occurs when the model predicts the positive class inaccurately. Meanwhile, a FN is an outcome in which the model classifies the negative class inaccurately.

Furthermore, other classification metrics, such as accuracy, F1-score, precision, and recall, were used to evaluate the models' performances. The accuracy, calculated using Equation 1, is the ratio of the number of correct predictions divided by the total number of input samples. Meanwhile, recall, calculated using Equation 2, measures the model's ability to detect positive samples. On the other hand, precision, calculated using Equation 3, measures the model's accuracy in classifying a sample as positive. Lastly, F1-score, which summarizes a model's predictive performance by combining two previously opposing variables — precision and recall, is calculated using Equation 4. F1-score could be considered the best metric in this study since the dataset has an uneven class distribution.

TABLE IV. TEXT PREPROCESSING RESULTS

Without Text Preprocessing	With Text Preprocessing	English Version
mohon bantuannya got **mampet**, sudah saya bersihkan sebagian kotorannya tapi masih **mampet** dan sampai mengalir ke jalan **#gotmampet #gotkotor**	mohon bantuannya got **mampat**, sudah saya bersihkan sebagian kotorannya tetapi masih **mampat** dan sampai mengalir ke jalan.	please help the gutter is clogged, i have cleaned some of the dirt but it is still clogged and the water flows into the street
jl Taman malaka selatan 3, duren sawit, jakarta timur	**jalan** taman malaka selatan tiga duren sawit jakarta timur	street of south malaka park three duren sawit east jakarta
laporan pak .di wilayah Cakung tepat nya di **keluarahan** rawa Terate kali sudah dangkal pak mohon untuk segera di keruk lumpur **bir dlm**.**sudh** mengadu ke lurah rawa Terate **ttp blm** ada tanggapan **smpi skrng**	laporan pak, di wilayah cakung tepatnya di **kelurahan** rawa terate kali sudah dangkal pak. mohon untuk segera di keruk lumpur **biar dalam**. **sudah** mengadu ke lurah rawa terate **tetap belum** ada tanggapan **sampai sekarang**.	sir, in the cakung area, to be precise, in the rawa terate sub-district, the river is already shallow. please immediately dredge the mud. i have complained to the village head, still no response until now
jln cipinang lontar **rt13rw06 deket** bekas hotel ahmad mas .. mohon solusi nya bapak ibu **yg** terhormat karena setiap hujan kami kebanjiran karena **GOT yg** susah untuk di bersihkan nya karena sebagian warga depan rumah nya itu di tinggikan di atas **GOT**	**jalan** cipinang lontar **rukun tetangga 013 rukun warga 006** dekat bekas hotel ahmad mas. mohon solusinya bapak ibu **yang** terhormat karena setiap hujan kami kebanjiran karena **got yang** susah untuk di bersihkannya karena sebagian warga depan rumahnya itu ditinggikan di atas **got**.	cipinang lontar street, rt 013 rw 006 near the former hotel ahmad. please provide a solution, because every time it rains we are flooded. the sewers are difficult to clean because the front part of some of the residents' houses are elevated above the sewers.
@TMCPoldaMetro @RadioElshinta @DKIJakarta Kerusakan penutup gorong-gorong yang sama yang diperbaiki **tgl** 5 maret lalu. Jelek banget kualitasnya, bikin macet panjang di pertigaan Tipar Cakung - RGTC sampai 1 KM **@DKIJakarta kok** tidak ditindak lanjuti kerusakan penutup gorong-gorong ini?	kerusakan penutup gorong-gorong yang sama yang diperbaiki **tanggal** 5 maret lalu. jelek banget kualitasnya, bikin macet panjang di pertigaan tipar cakung rgtc sampai satu kilometer, **ko** tidak ditindak lanjuti kerusakan penutup gorong-gorong ini?	same damage to the culvert cover which was repaired on 5 march. very bad quality, causing a long traffic jam at the tipar cakung rgtc junction for up to one kilometer, how come no action is taken to fix the damage on this culvert cover?
Pak, mohon dibuatkan sumur resapan disepanjang jalan ini, selalu rawan **banjirrrrrr**	pak, mohon dibuatkan sumur resapan di sepanjang jalan ini, selalu rawan **banjir**.	sir, please make an infiltration well along this road, it is always prone to flooding.

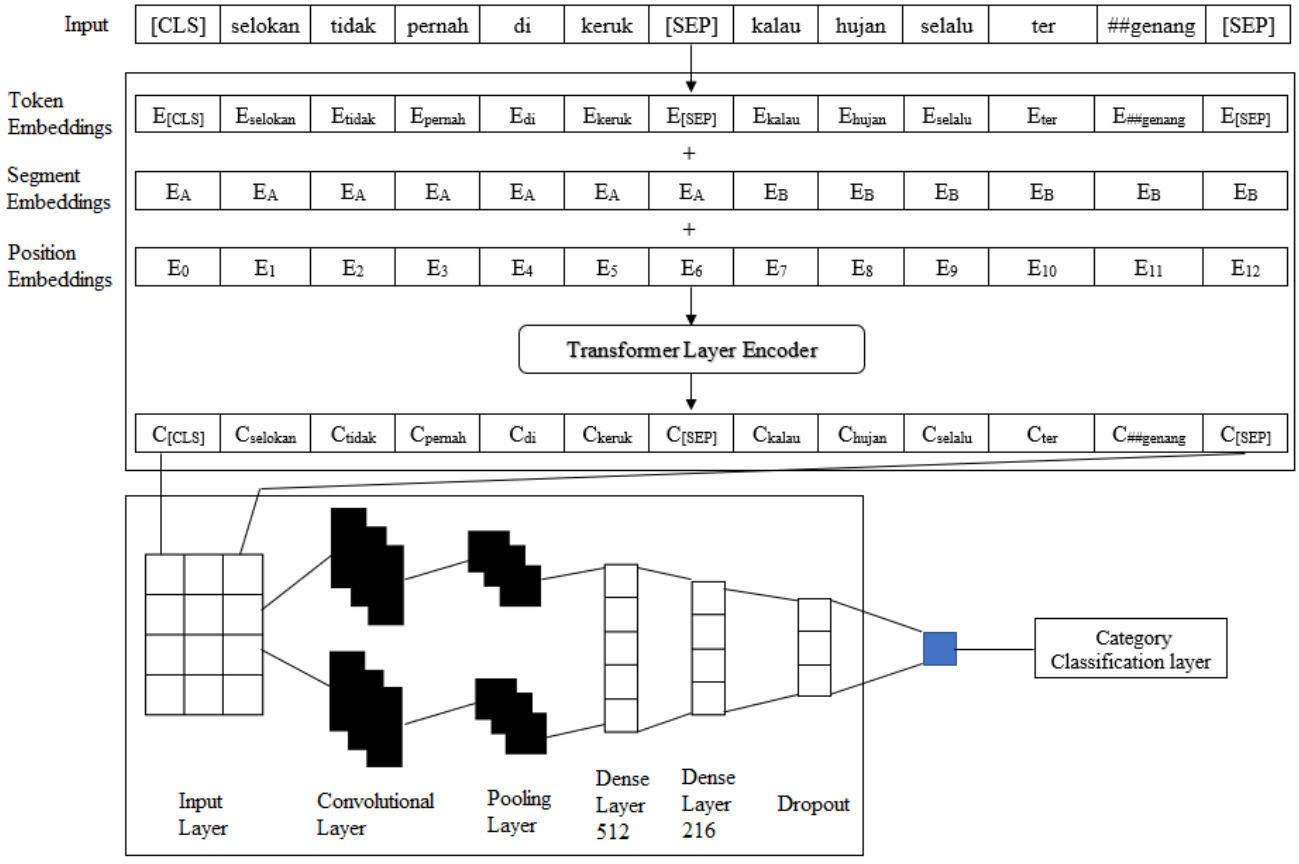

Fig. 2. Representation of BERT Embeddings and CNN.

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

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

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

$$F1 - score = \frac{2 \times precision \times recall}{precision + recall} \quad (4)$$

Finally, the performance results of DNN models with and without text processing were compared to see if there is a significant difference between both approaches using a statistical t-Test, Paired Two Sample for Means. The test statistic is calculated using Equation 5 where $\bar{d}$ is the sample mean of the differences, s is the sample standard deviation of the differences, n is the sample size, and t is the Student t quantile with n-1 degrees of freedom used to define the p-value. In statistical significance testing, it is the likelihood of receiving a test statistic at least as extreme as the observed one, assuming that the null hypothesis is true. The null hypothesis is rejected when the p-value is smaller than the predetermined significance level, suggesting that the observed result would be highly implausible under the null hypothesis.

$$t = \frac{\bar{d}}{s/\sqrt{n}} \quad (5)$$

III. Results

Some of the results of text data after processing with and without text preprocessing used in building models are shown in Table IV. The performance results of various experiments without text preprocessing are shown in Table V, where the highest values are printed bold. The fine-tuned DNN architecture with IndoBERT base model to text data of society reports without text preprocessing produced the best accuracy and precision using the CNN algorithm model architecture with a learning rate of $3x10^{-5}$ and a batch size of 16. The best accuracy and precision obtained were 83.85% and 88.76%, respectively. However, the best result on F1-score and recall was achieved using a learning rate of $5x10^{-5}$ and a batch size of 16. The best F1-score and recall yielded 85.44% and 83.95%, respectively.

Meanwhile, the performance results of various experiments with text preprocessing are represented in Table VI, where the best values are boldly printed. The fine-tuned deep learning model with text preprocessing obtained the best accuracy, F1-score, and recall using the CNN algorithm with a learning rate of $5x10^{-5}$ and batch size of 16. The best accuracy, F1-score, and recall yielded 84.47%, 85.92%, and 85.54%, respectively. However, the best precision of 87.37% was produced using a learning rate of $2x10^{-5}$ and a batch size of 32.

From the two tables, it can be seen that the CNN algorithm produced the best performances compared to other algorithms. CNN even produced better results using text preprocessing. With a learning rate of $5x10^{-5}$ and a batch size of 16, the accuracy, F1-score, precision, and recall were improved by 1.45%, 0.48%, 0.06%, and 1.59%, respectively. However, these values indicated that the text preprocessing made minimal performance improvements.

Moreover, in order to make statistical comparisons of performance results of fine-tuned DNN model between with and without text preprocessing, the t-Test was performed. The statistic test can determine whether the differences between two approaches are significant. If the p-value is less than a significant level, the null hypothesis, which states no significant differences between the models, is rejected. The results are shown in Table VII, where p-values less than a significant level of 0.05, which indicated significant differences, are printed bold. It can be seen that text preprocessing had no significant effect on most architectures, such as in CNN, LSTM, and Bi-LSTM. Meanwhile, the text preprocessing significantly affected the performances of GRU as well as the accuracy and the precision of MLP but not the F1-score.

IV. Discussion

The result of insignificant performance differences between DNN architectures with and without text preprocessing is in line with the study conducted by [22]. It performed various experiments with three preprocessing tasks: lowercasing, lemmatizing, and multiword grouping. It concluded that simple tokenization was generally adequate in DNN architectures, particularly CNN. Furthermore, the simple preprocessing worked equally or better than more complex techniques in most cases, except for domain-specific texts, such as in the medical field, where apparent differences were needed to classify cardiovascular disease.

We proved that the study conducted by [20] claimed that removing noises in the text data would significantly improve the accuracy of a pre-trained BERT model could be gone wrong. Instead, a BERT model applied to heavily cleaned text data could make things worse because the contextual information would be lost. This finding is supported by a study conducted by [23] when profiling authors from their writings. When no preprocessing method was used, the study found that BERT best predicted an author's gender. In the best scenario, the model was 86.67% accurate in estimating the writers' gender.

One possible reason for the excellent performance of the model without text preprocessing in this study was the suitable choice of the word embeddings approach. The IndoBERT was trained on an extensive Indonesian text dataset that includes formal and slang language, such as tweets. The results could differ if the pre-trained BERT models were trained using Wikipedia corpus only, like the BERT-Base multilingual pre-trained model [16]. A study conducted in [19] showed that preprocessing steps and further preprocessing processes were needed when using BERT multilingual pre-trained model to improve the classification performance of a DNN. The more data BERT pre-train, the less the negative impact of misspellings and slang words would be because the model has more examples of typos, abbreviated vocabularies, orthographic errors, et cetera.

Furthermore, CNN managed to perform best when utilizing the BERT embeddings compared to the other DNN architec-

TABLE V. PERFORMANCE RESULTS OF MODELS WITHOUT TEXT PREPROCESSING

Architecture	Learning Rate	Batch Size	Accuracy (%)	F1-Score (%)	Precision (%)	Recall (%)
MLP	$5x10^{-5}$	16	**78.26**	60.49	66.62	56.87
	$3x10^{-5}$	16	77.02	60.43	**67.04**	56.02
	$2x10^{-5}$	16	76.40	61.24	66.47	57.58
	$5x10^{-5}$	32	76.81	**62.95**	65.71	**61.08**
	$3x10^{-5}$	32	76.19	58.19	66.86	53.88
	$2x10^{-5}$	32	73.91	59.56	65.92	54.98
LSTM	$5x10^{-5}$	16	69.98	54.17	54.93	54.09
	$3x10^{-5}$	16	72.26	**65.14**	**77.95**	**61.06**
	$2x10^{-5}$	16	69.36	52.68	57.14	50.20
	$5x10^{-5}$	32	**72.67**	55.91	58.63	54.20
	$3x10^{-5}$	32	72.26	56.38	57.16	55.87
	$2x10^{-5}$	32	71.01	53.07	55.30	51.40
BiLSTM	$5x10^{-5}$	16	**81.37**	79.87	**87.92**	74.74
	$3x10^{-5}$	16	79.71	**83.19**	84.84	**82.15**
	$2x10^{-5}$	16	80.54	73.58	84.01	69.85
	$5x10^{-5}$	32	81.16	79.99	86.28	75.56
	$3x10^{-5}$	32	80.33	72.34	86.02	67.90
	$2x10^{-5}$	32	79.50	76.09	82.67	72.10
CNN	$5x10^{-5}$	16	83.02	**85.44**	87.20	**83.95**
	$3x10^{-5}$	16	**83.85**	75.56	**88.76**	70.15
	$2x10^{-5}$	16	80.12	79.33	86.86	73.99
	$5x10^{-5}$	32	82.61	80.19	86.65	75.73
	$3x10^{-5}$	32	79.92	72.41	85.84	66.93
	$2x10^{-5}$	32	80.54	73.27	87.29	67.17
GRU	$5x10^{-5}$	16	72.05	65.44	75.55	62.15
	$3x10^{-5}$	16	**74.33**	**68.36**	**80.40**	64.28
	$2x10^{-5}$	16	72.26	66.01	79.09	60.98
	$5x10^{-5}$	32	72.05	67.26	76.99	**65.57**
	$3x10^{-5}$	32	74.12	57.74	61.37	54.99
	$2x10^{-5}$	32	70.81	56.67	60.62	54.36

TABLE VI. PERFORMANCE RESULTS OF MODELS WITH TEXT PREPROCESSING

Architecture	Learning Rate	Batch Size	Accuracy (%)	F1-Score (%)	Precision (%)	Recall (%)
MLP	$5x10^{-5}$	16	74.74	59.55	63.70	56.61
	$3x10^{-5}$	16	74.95	60.20	63.39	57.43
	$2x10^{-5}$	16	73.91	59.27	65.61	54.93
	$5x10^{-5}$	32	75.36	**60.43**	63.86	**58.34**
	$3x10^{-5}$	32	**76.40**	59.60	**66.27**	55.93
	$2x10^{-5}$	32	73.08	58.44	64.25	54.29
LSTM	$5x10^{-5}$	16	68.32	51.08	53.48	50.20
	$3x10^{-5}$	16	**73.50**	57.67	59.59	**56.18**
	$2x10^{-5}$	16	70.19	54.02	57.58	52.07
	$5x10^{-5}$	32	69.98	**61.68**	**79.68**	55.77
	$3x10^{-5}$	32	71.22	55.08	57.27	54.62
	$2x10^{-5}$	32	68.32	52.42	54.58	50.74
BiLSTM	$5x10^{-5}$	16	**80.75**	**79.32**	83.81	**76.47**
	$3x10^{-5}$	16	79.30	78.14	82.33	75.37
	$2x10^{-5}$	16	80.33	74	**85.77**	69.58
	$5x10^{-5}$	32	79.50	78.70	84.25	75.27
	$3x10^{-5}$	32	79.50	72.02	83.10	68.04
	$2x10^{-5}$	32	80.33	78.59	83.12	76.16
CNN	$5x10^{-5}$	16	**84.47**	**85.92**	87.26	**85.54**
	$3x10^{-5}$	16	82.61	74.59	85.14	70.83
	$2x10^{-5}$	16	81.57	80.71	86.60	76.56
	$5x10^{-5}$	32	83.64	82.40	87.35	79
	$3x10^{-5}$	32	83.23	65.75	68.06	64.39
	$2x10^{-5}$	32	80.95	74.43	**87.37**	68.86
GRU	$5x10^{-5}$	16	**72.67**	**66.35**	**78.86**	**61.16**
	$3x10^{-5}$	16	71.64	56.96	58.05	56
	$2x10^{-5}$	16	71.43	56.67	59.42	54.64
	$5x10^{-5}$	32	71.84	57.40	59.34	55.88
	$3x10^{-5}$	32	70.60	56.60	58.70	55.16
	$2x10^{-5}$	32	69.15	54.67	57.10	52.87

TABLE VII. P-VALUES OF SIGNIFICANT TEST RESULTS

Architecture	Accuracy	F1-Score	Precision	Recall
MLP	**0.0124 (sig.)**	0.8727 (not sig.)	**0.00519 (sig.)**	0.29054 (not sig.)
LSTM	0.10471 (not sig.)	0.31956 (not sig.)	0.48675 (not sig.)	0.16681 (not sig.)
BiLSTM	0.10323 (not sig.)	0.25638 (not sig.)	0.10761 (not sig.)	0.43984 (not sig.)
CNN	0.06955 (not sig.)	0.38748 (not sig.)	0.14476 (not sig.)	0.10301 (not sig.)
GRU	**0.04053 (sig.)**	**0.02643 (sig.)**	**0.03164 (sig.)**	**0.02431 (sig.)**

tures. It could be because the CNN algorithm might extract local and global features very well from the vectors using the convolutional, the pooling, and the fully connected (dense) layers, which can maintain semantic context meaning on text data. This finding supports studies on sentiment analysis of a commodity review and stance detection for credibility analysis of information on social media conducted by [24], [25]. These studies showed that BERT embeddings and CNN obtained better results than single CNN that ignores relation contextual semantics on text.

BERT's input embeddings are composed of three different embeddings: Token Embeddings, Segment Embeddings, and Position Embeddings. Before being passed to the Token Embeddings layer, the input text is tokenized using a method called WordPiece tokenization. It is a data-driven tokenization strategy that balances vocabulary size and OOV words. Extra tokens are also added to the beginning and the end, namely the classification token ([CLS]) and the NSP token ([SEP]). These tokens have two functions: one serves as a representation input for classification tasks, and the other is to split a pair of input texts. Then, the sentence number is converted into a vector in Segment Embeddings. Meanwhile, the Position Embeddings create a vector for the word's position within the sentence. Finally, the three embeddings are summed up to generate a single shape representation passed to BERT's encoder layer [9]. Our proposed architecture is shown in Fig. 2.

V. CONCLUSION

This study showed that most text preprocessing tasks such as formalizing slang, fixing typos, case-folding, et cetera were not absolute things to do with transfer learning if the word embeddings method and DNN architecture were chosen correctly. There were insignificant differences between models with or without text preprocessing on most DNN architectures such as LSTM, Bi-LSTM, and CNN when utilizing BERT embeddings. Furthermore, combining BERT embeddings and CNN produced the best classification performance. Rather than wasting time preprocessing text data, researchers should focus on finding a suitable word embeddings approach and DNN architecture. Future studies should investigate the significance of each text preprocessing step since there were significant differences in performance results between the model with and without text preprocessing using GRU and MLP.

REFERENCES

[1] A. F. Hidayatullah, S. Cahyaningtyas, and A. M. Hakim, "Sentiment analysis on twitter using neural network: Indonesian presidential election 2019 dataset," *IOP Conference Series: Materials Science and Engineering*, vol. 1077, no. 1, p. 012001, feb 2021, doi: 10.1088/1757-899x/1077/1/012001.

[2] L. P. Manik, H. Febri Mustika, Z. Akbar, Y. A. Kartika, D. Ridwan Saleh, F. A. Setiawan, and I. Atman Satya, "Aspect-based sentiment analysis on candidate character traits in indonesian presidential election," in *2020 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*, 2020, pp. 224–228, doi: 10.1109/ICRAMET51080.2020.9298595.

[3] E. Haddi, X. Liu, and Y. Shi, "The role of text pre-processing in sentiment analysis," *Procedia Computer Science*, vol. 17, pp. 26–32, 2013, first International Conference on Information Technology and Quantitative Management, doi: 10.1016/j.procs.2013.05.005.

[4] P. P. Adikara, S. Adinugroho, and S. Insani, "Detection of cyber harassment (cyberbullying) on instagram using naïve bayes classifier with bag of words and lexicon based features," in *Proceedings of the 5th International Conference on Sustainable Information Engineering and Technology*, ser. SIET '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 64–68, doi: 10.1145/3427423.3427436.

[5] M. A. Fauzi, "Word2vec model for sentiment analysis of product reviews in indonesian language," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 1, p. 525, Feb. 2019, doi: 10.11591/ijece.v9i1.pp525-530.

[6] R. P. Nawangsari, R. Kusumaningrum, and A. Wibowo, "Word2vec for indonesian sentiment analysis towards hotel reviews: An evaluation study," *Procedia Computer Science*, vol. 157, pp. 360–366, 2019, the 4th International Conference on Computer Science and Computational Intelligence (ICCSCI 2019) : Enabling Collaboration to Escalate Impact of Research Results for Society, doi: 10.1016/j.procs.2019.08.178.

[7] L. P. Manik, A. Ferti Syafiandini, H. F. Mustika, A. Fatchuttamam Abka, and Y. Rianto, "Evaluating the morphological and capitalization features for word embedding-based pos tagger in bahasa indonesia," in *2018 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, 2018, pp. 49–53, doi: 10.1109/IC3INA.2018.8629519.

[8] J. Abdillah, I. Asror, and Y. F. A. Wibowo, "Emotion classification of song lyrics using bidirectional LSTM method with GloVe word representation weighting," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 4, no. 4, pp. 723–729, Aug. 2020, doi: 10.29207/resti.v4i4.2156.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186, doi: 10.18653/v1/N19-1423.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[11] H. Tian, K. Yang, D. Liu, and J. Lv, "Anchibert: A pre-trained model for ancient chinese language understanding and generation," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–8, doi: 10.1109/IJCNN52387.2021.9534342.

[12] D. Q. Nguyen and A. Tuan Nguyen, "PhoBERT: Pre-trained language models for Vietnamese," in *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020, pp. 1037–1042, doi: 10.18653/v1/2020.findings-emnlp.92.

[13] B. Wilie, K. Vincentio, G. I. Winata, S. Cahyawijaya, X. Li, Z. Y. Lim, S. Soleman, R. Mahendra, P. Fung, S. Bahar, and A. Purwarianti, "IndoNLU: Benchmark and resources for evaluating Indonesian natural language understanding," in *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*. Suzhou, China: Association for Computational Linguistics, Dec. 2020, pp. 843–857.

[14] 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 International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 757–770, doi: 10.18653/v1/2020.coling-main.66.

[15] L. P. Manik, Z. Akbar, H. F. Mustika, A. Indrawati, D. S. Rini, A. D. Fefirenta, and T. Djarwaningsih, "Out-of-scope intent detection on a knowledge-based chatbot," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 5, pp. 446–457, Oct. 2021, doi: 10.22266/ijies2021.1031.39.

[16] K. S. Nugroho, A. Y. Sukmadewa, H. Wuswilahaken DW, F. A. Bachtiar, and N. Yudistira, *BERT Fine-Tuning for Sentiment Analysis on Indonesian Mobile Apps Reviews*. New York, NY, USA: Association for Computing Machinery, 2021, p. 258–264, doi: 10.1145/3479645.3479679.

[17] I. F. Putra and A. Purwarianti, "Improving indonesian text classification using multilingual language model," in *2020 7th International Conference on Advance Informatics: Concepts, Theory and Applications (ICAICTA)*, 2020, pp. 1–5, doi: 10.1109/ICAICTA49861.2020.9429038.

[18] J. Fawaid, A. Awalina, R. Y. Krisnabayu, and N. Yudistira, *Indonesia's Fake News Detection Using Transformer Network*. New York, NY, USA: Association for Computing Machinery, 2021, p. 247–251, doi: 10.1145/3479645.3479666.

[19] M. R. Yanuar and S. Shiramatsu, "Aspect extraction for tourist spot review in indonesian language using bert," in *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 2020, pp. 298–302, doi: 10.1109/ICAIIC48513.2020.9065263.

[20] W. Maharani, "Sentiment analysis during jakarta flood for emergency responses and situational awareness in disaster management using bert," in *2020 8th International Conference on Information and Communication Technology (ICoICT)*, 2020, pp. 1–5, doi: 10.1109/ICoICT49345.2020.9166407.

[21] A. R. Isnain, A. Sihabuddin, and Y. Suyanto, "Bidirectional long short term memory method and word2vec extraction approach for hate speech detection," *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 14, no. 2, p. 169, Apr. 2020, doi: 10.22146/ijccs.51743.

[22] J. Camacho-Collados and M. T. Pilehvar, "On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 40–46, doi: 10.18653/v1/W18-5406.

[23] E. Alzahrani and L. Jololian, "How different text-preprocessing techniques using the bert model affect the gender profiling of authors," in *Advances in Machine Learning*. Academy and Industry Research Collaboration Center (AIRCC), Sep. 2021, doi: 10.5121/csit.2021.111501.

[24] J. Dong, F. He, Y. Guo, and H. Zhang, "A commodity review sentiment analysis based on bert-cnn model," in *2020 5th International Conference on Computer and Communication Systems (ICCCS)*, 2020, pp. 143–147, doi: 10.1109/ICCCS49078.2020.9118434.

[25] H. Karande, R. Walambe, V. Benjamin, K. Kotecha, and T. Raghu, "Stance detection with BERT embeddings for credibility analysis of information on social media," *PeerJ Computer Science*, vol. 7, p. e467, Apr. 2021, doi: 10.7717/peerj-cs.467.