

Enhancing *Hate speech* Detection : Leveraging *Emoji* Preprocessing With *LSTM* Models

Junita Amalia, Yoga Sihombing, Gabriel Saputra Panggabean, Mares Siagian

¹Information Systems, Institut Teknologi Del, Laguboti, Indonesia

Article Info

Article history:

Received month dd, yyyy

Revised month dd, yyyy

Accepted month dd, yyyy

Keywords:

Text Classification

LSTM

Emoji Description

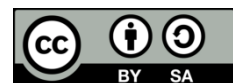
Emoji Embedding

Tweet

ABSTRACT

Social media, especially X (formerly Twitter), has become an important platform for online communication, allowing the exchange of ideas and views across geographical and time boundaries. However, tweets often lack structure, making it difficult to identify important messages. The unregulated nature of communication on social media also makes it a breeding ground for hate speech, which can have harmful effects on individuals and communities. Therefore, the development of technologies such as emotion identification through emojis is crucial not only to understand users' emotional expressions but also to identify and reduce the spread of hate speech. In recent years, *hate speech* detection has become an important area of research due to the rise of toxic and harmful content online. By recognizing patterns that indicate hateful or harmful language, we can mitigate the negative impact of these messages. This research focuses on implementing *Long Short Term Memory (LSTM)* to classify text in the form of tweets. *LSTM* is capable of capturing long-range dependencies in sequential data, maintaining and streaming information through "cell state" memory units. Through this model, it becomes possible to detect nuanced expressions of emotions while simultaneously identifying hate speech, contributing to the improvement of online discourse. This study aims to analyze the effect of *emoji description* and *emoji embedding* in comparison with *delete emoji* method on *hate speech* classification. The *LSTM + Emoji Description* model achieved an accuracy of 0.924 and the *LSTM + Emoji Embedding* model achieved an accuracy of 0.917. While, the *LSTM + Delete Emoji* model only achieved an accuracy of 0.68. This difference in accuracy indicates that emojis have an important role in the classification of hate speech, especially in datasets that focus on emojis to determine the label.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Gabriel Saputra Panggabean

Information System, Institut Teknologi Del

Laguboti, Indonesia

Email: gabrielpanggabean1212@gmail.com

1. INTRODUCTION

Social media, especially X (Twitter), has become a significant platform for communicating online, transcending geographic and time barriers. X (Twitter) is a micro blogging site operated by Twitter, Inc. Called micro blogging because this site allows users to send and read messages like blogs in general [1]. X (Twitter) users can share their thoughts, interests, opinions, and sentiments on various topics and issues they encounter in their daily lives [2]. Sentiment analysis on X (Twitter) has become a popular research topic. Sentiment

analysis is useful in the world of marketing and social studies, as it can provide insights into users' opinions and sentiment expressions [3].

Hate comment is one of the bad communication patterns because it contains elements of insults made in the comments column of social media accounts [4]. *Hate speech* aims to damage the reputation of the targeted person and cause discord among social media users [5]. Using offensive language can be harmful and can cause emotional distress to the targeted person or group [6]. *Hate speech* refers to the behavior of individuals or groups that express defamation, insult, provocation, defamation, or other actions with the aim of harming, provoking violence, causing social conflict, or discriminating. While X (Twitter) boasts millions of active users who generate different types of content every day, the presence of *hate speech* on it poses challenges in maintaining a safe and inclusive environment [7]. So many tweets apply implicit hate, emojis, and slang words to express hate speech [8]. Emojis are signs or images often used on the internet, text messages, and social media to express emotions or ideas. Emojis were created so that users can use something that can be fully controlled by the sender of the message [9].

In the research "Detection of *Hate speech* and Offensive Language in Twitter Data Using *LSTM* Model", some limitations of the basic model, such as imbalance of result data and accuracy improvement especially for *LSTM* and *Bi-LSTM* models, were overcome. The results showed that the *LSTM* model achieved the highest accuracy of 86%, while the *Bi-LSTM* model achieved an accuracy value of 80.1%. This research contributes to overcoming classifier constraints by introducing multi-step strategies and word embedding through deep neural network [10].

The study entitled "*Hate speech* Detection on Twitter through Natural Language Processing using *LSTM* Model" discusses *hate speech* detection on twitter using natural language processing and a deep learning model called Long Short-Term Memory (*LSTM*). The results show that the *LSTM-FastText* model with random oversampling has better performance in classifying *hate speech* on X (Twitter) in Indonesian. The *LSTM-FastText* model achieved an F1-score of 89.91%, while the *LSTM-GloVe* model achieved an *F1-Score* of 82.14%. So, it can be seen that the *LSTM-FastText* model has a higher F1-score than the *LSTM-GloVe* model [11].

The research "*Emoji Description* Improves Tweet Classification" utilizes emojis that are usually omitted during text classification by replacing emojis with their text descriptions and using pre-trained word embeddings commonly called EMJ-DESC. In the processing, only 10 frequently used emojis are used. In the study, the EMJ-EMBED method had an accuracy of 78.64%. While the EMJ-DESC method has an accuracy of 80.56%. The accuracy optimization of EMJ-EMBED comes from EMJ-EMBED replacing emoji with vectors of emoji and using pre-trained word embeddings. In this method, both word embeddings and *emoji embeddings* are used to represent the text. EMJ-DESC, on the other hand, only replaces emojis with their text descriptions without using *Emoji Embeddings* [5].

Based on the research that has been presented, there has been no discussion of how an emoji is processed so that researchers try to see the influence of emojis for the classification of tweets that have emojis. This study aims to evaluate the influence of emoji in the *hate speech* classification process using the *LSTM* model, compared to the emoji removal method at the text pre-processing stage. This research resulted in three different models: *LSTM* model with *Delete Emoji*, *LSTM* model with *Emoji Description*, and *LSTM* model with *Emoji Embedding*. In addition, this research uses all emojis in the dataset, unlike the previous research which only used 10 emojis. Therefore, this study aims to analyze the effect of emoji desc and emoji embedding on hate speech classification.

2. METHOD

Methodology is a description that contains the stages carried out during this research. This description is needed as a reference so that the results obtained meet the objectives of this research. This research has several stages that must be fulfilled before continuing with other stages. The stages in the research include :

2.1. Dataset

The dataset used was taken from *HuggingFace* [12]. *HuggingFace* is a community platform that provides various resources in the field of natural language processing NLP (Natural Language Processing). This dataset has 4743 rows for train *hate speech* data which has been labeled with 11 attributes, test data there are 594 rows, and for validation data there are 592 row with 11 attributes. After augmentation data, data will increase from 5677 data to 30.500 data. In this research, the attributes that will be used are *text* and *label_gold*. *Text* contains the text and emoji that are being classified, so the text column is a very important feature to classify hate speech. classification of *hate speech*. *Label_gold* is used to determine whether the text is *hate speech* or non-*hate speech*. Based on the analysis that has been done, the dataset used is balanced with the percentage of *hate speech* totaling 50.48% and non-*hate speech* 49.51%. This research will use train data 80%, 20% test data, and 10% validation data taken from train data using *k-fold cross-validation*.

2.2. Data Preprocessing

Data preprocessing is a process or stage that is conducted with the purpose of managing incoming raw data and converting it into superior data or optimal input to proceed to the next step [13]. Data preprocessing needs to be implemented so that the data can be used for the machine learning training process that will be built. This dataset still requires *preprocessing* before being used as a feature to classify *hate speech*.

2.2.1. Data Augmentation

Data augmentation is a technique used to overcome the limited number of datasets. *Data augmentation* purposes to improve the performance of machine learning models by reducing the occurrence of *overfitting* [14]. *Data augmentation* is also a way utilized for *text processing* and machine learning to increase the amount and variety of training data without the need to collect new data [15]. *Data augmentation* is also identified as a way to reduce *overfitting* [16].

2.2.2. Data Cleaning

Data cleaning is a process of cleaning incomplete data values, correcting data inconsistencies, and minimizing noise when identifying *outliers* [17].

2.2.3. Drop Column

Drop column in this research is needed to remove attributes that are not used in *hate speech* classification. The attributes used in this research are text and label_gold.

2.3. Text Preprocessing

Text preprocessing is used to convert unstructured text into structured text so that it is easier to classify text [18]. Text preprocessing includes the stages of *emoji description*, *emoji embedding*, *remove hashtags*, *remove numbering*, *remove punctuation*, *case folding*, *tokenizing*, *remove stopwords*, *stemming*, and *text normalization*.

2.3.1. Emoji Description

Emoji description is a strategy of replacing emoji with their textual decryption. *Emoji description* will remove all emojis in the input and combine a fairly detailed decryption with some tokens from the emoji [19]. This process starts by replacing each emoji in the text with a text description detailing the meaning and expression of the emoji. For example, the emoji "😊" can be replaced with the text description "smile with smiling eyes". After this replacement, the emoji text description is converted into a vector representation using pre-trained word embeddings.

2.3.2. Emoji Embedding

Emoji embedding works by representing an emoji in the form of a numeric vector that captures the emotional content of the emoji. An example is the pairing of the emoji "😊" (smile) and the word "happy". Next, for each pair of data, a vector of description words is calculated using embedding methods such as word2vec embeddings. The word vectors are then summed to get a vector representation of the emoji. *Emoji embedding* works by generating a numerical representation of the emoji used in *hate speech* [20].

2.3.3. Case Folding

Case folding is one of the heuristics used to convert uppercase letters into lowercase letters. Case folding needs to be done to facilitate the data classification process such as reducing variations in the same word [21].

2.3.4. Remove Hashtag

Remove hashtag is done for several reasons, such as to change the focus of a particular topic, correct typos, or adapt the post to a different platform [22].

2.3.5. Remove Numbering

Remove numbering is a process where single numbers and letters in the dataset are removed [22]. This process can also improve the effectiveness of machine learning models, especially if the number or numbers are irrelevant to the task to be performed.

2.3.6. Remove Punctuation

Remove punctuation is the process of cleaning data by removing punctuation marks and symbols because the machine does not recognize punctuation marks. *Punctuation* marks such as commas, periods, exclamation marks, etc. do not always provide significant information in some text processing tasks and can interfere with analysis [22].

2.3.7. Tokenizing

Tokenizing speeds up the data classification process because it no longer requires breaking the text manually. Tokenizing can identify patterns in the text faster, because tokenizing breaks the text into words or phrases [22].

2.3.8. Remove Stopword

Remove stopwords is a step to remove words or tokens that do not provide important information from a text. The words or tokens that are removed are repeated words, such as is, am, are and other words [22].

2.3.9. Stemming

Stemming is used as a process that will be utilized in normalization of words based on the basic form which is lemma form. The data used at this stage is data that has been preprocessed from the previous stage, namely from the results of removing stop words. If an affix is found in a word token, the word will be converted into a word basically [23].

2.3.9. Text Normalization

Text normalization is the process of converting informal text into a more standard and consistent form [24]. *Text normalization* is used to manage slang words in the dataset to ensure that the *LSTM* model can understand and classify the text more accurately. The authors conducted the analysis manually by checking one by one the slangwords contained in the data, and then normalized the slangwords by replacing them with words that fit the normal language. Therefore, it can be ensured that all slangwords have been handled properly and have been normalized appropriately.

2.4. LSTM

LSTM (Long Short Term Memory) has the ability to learn and capture long-term dependencies in text by using gates, which are internal mechanisms that regulate the flow of information in a network. *LSTM* is used to analyze text containing *hate speech*. The *LSTM* model is trained using a *hate speech* dataset, where the model learns to recognize patterns and features that indicate the presence of *hate speech*. By utilizing the nature of sequences in comments, *LSTM* can effectively capture context and dependencies between words or characters, thus enabling accurate *hate speech* detection [12]. The main layers used to create the *LSTM* network architecture include embedding layer, *LSTM* layer, dropout layer, and dense layer. In this study, a stacked *LSTM* architecture is used, where multiple *LSTM* layers are stacked on top of each other. This architecture allows the model to capture more complex patterns and dependencies in sequential data, thus improving the model's ability to accurately detect *hate speech*. Here is the architecture of the *LSTM* model used. Researchers utilized a stacked LSTM-based architecture in this model. The selection of LSTM and several additional layers was based on a series of experiments conducted previously. Through these experiments, the researchers successfully identified the optimal configuration to achieve the best performance for the designated task, as illustrated in the figure 1.

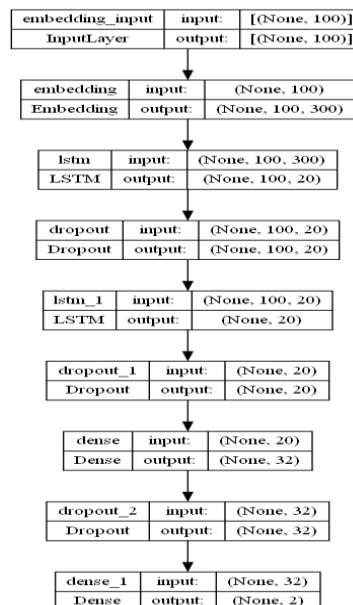


Figure 1. *LSTM* Model Architecture

2.4.1. Embedding Layer

Embedding layer used to define the input and output of the vector dimensions that will be carried out by the machine learning modeling that will be used.

2.4.2. LSTM Layer

At the *LSTM layer*, there is one *LSTM* unit that functions as a forward *layer*. This *layer* is responsible for storing information from the input sequence starting from the front to the back. By using only one direction, namely forward, *LSTM* is able to capture temporal relationships in sequence data. Even though it does not read input backwards, the *LSTM* model is still effective in understanding the meaning and context of a sentence.

The output of this *LSTM layer* provides a vector representation that reflects each object in the sequence. *LSTM* model formation can be implemented using the Keras library. Researchers use parameters to optimize the *LSTM* model [25].

2.4.3. Dropout Layer

Dropout layer neural networks are used to overcome overfitting problems by randomly removing some neurons in the hidden or visible *layer*. Dropout selects some neurons with a certain probability and temporarily removes them from the network.

2.4.4. Dense Layer

Dense layer used to perform operations on the input *layer*. This *layer* has the ability to add non-linear properties, thus providing the opportunity to model various mathematical functions. The *dense layer* implementation involves several operations, including an element-wise activation function passed as an activation argument, a weight matrix (kernel) created by the *layer*, and a bias vector (if *use_bias* is set to true).

2.5. K-Fold Cross-Validation

K-Fold Cross-Validation is a model performance evaluation method that divides data into *folds*, then trains the model *k* times using *k-1 folds* as training data and the remaining *folds* as validation data [10]. The *K-Fold Cross-Validation* used by researchers is *2-fold cross-validation* because it shows that the use of *2-fold* produces more stable and better results than *5-fold cross-validation*.

2.6. Evaluation Model

In this research, 3 experiments conducted, where each model will run 3 text classification experiments with emoji, namely *LSTM + Delete Emoji*, *LSTM + Emoji Description*, and *LSTM + Emoji Embedding*. Then, the *accuracy*, *recall*, *precision*, *recall*, and *f1-score* metrics will be used to evaluate the model. The four metrics are used for the reason of determining how accurate the model built is.

3. RESULTS AND DISCUSSION

3.1. Test Result

Testing is carried out to determine the most optimal model. Before testing, data that has passed the *preprocessing* stage has been separated into three, namely train data, test data, and validation data with a comparison ratio of 80% : 20% : 10%. Researchers will implement emoji preprocessing, where emoji preprocessing is a series of steps or techniques applied to the dataset to prepare and manage the use of emoji before being applied to other natural language processing or text analysis models. The emoji preprocessing implementation is *LSTM + Delete Emoji*, *LSTM + Emoji Description*, and *LSTM + Emoji Embedding*. In the *LSTM + Delete Emoji* experiment, the emoji in the text will be deleted for comparison with models that do special processing for emoji so that the effect of emoji can be seen by comparing the implementation of delete emoji with the other two models that do emoji preprocessing. The *LSTM + Emoji Description* experiment will convert the emoji in the text into words. This function uses the *demojize* method from the emoji library which converts each emoji in the input string into its text representation. The *LSTM + Emoji Embedding* experiment does not remove emoji in preprocessing, so the emoji will be vectorized along with the text so that the preprocessing stage is not removed emoji.

In this research, we will test each model, namely *LSTM + Delete Emoji*, *LSTM + Emoji Description*, and *LSTM + Emoji Embedding*. Two classes that will be used for classification, namely *hate speech (label 1)* dan *non-hate speech (label 0)*.

Table 1. Result Model

Evaluation Matrix	Result Model		
	<i>LSTM + Delete Emoji</i>	<i>LSTM + Emoji Description</i>	<i>LSTM + Emoji Embedding</i>
<i>Precision</i>	0,675	0,920	0,917
<i>Recall</i>	0,677	0,920	0,917
<i>F1-Score</i>	0,676	0,924	0,917
<i>Accuracy</i>	0,678	0,924	0,917

Base on that tabel, it was found that the comparison of *precision*, *recall*, *f1-score* and *accuracy* values between the *LSTM + Delete Emoji* model, *LSTM + Emoji Description* model, and *LSTM + Emoji Embedding*.

The experimental results show that emojis play an important role in text classification. Removing emoji without equivalent information replacement will damage the performance of the model. The *LSTM + Delete Emoji* model produces a relatively low *accuracy* of 0.678, *f1-score* 0.676, *recall* 0.677, *precision* 0.675 is a relatively low result for classification results. This shows that important information carried by emoji is lost, so the model is not able to recognize both labels well. In comparison, using emoji description and emoji embedding improved the overall *accuracy*, *f1-score*, *recall*, and *precision*. Both methods retain essential emotional and contextual information, which helps the model make more accurate and balanced predictions. The use of *emoji description* increased *accuracy* to 0.924, *f1-score* 0.924, *recall* 0.92, and *precision* 0.92. Similarly, *emoji embedding* resulted in *accuracy* 0.917, *f1-score* 0.917, *recall* 0.917, and *precision* 0.917. So from the classification results, it can be seen that the approach using *emoji description* is better than the other three approaches.

3.2 Discussion

Results from experiments show that emojis have an important role in text classification. Deleting emoji without equivalent information replacement significantly impairs model performance. The *LSTM + Delete Emoji* model produces relatively low evaluation metrics compared to the *LSTM + Emoji Description* and *LSTM + Emoji Embedding* models. The following is a visualization of the calculation results for *accuracy*, *precision*, *recall*, and *f1-score* values.

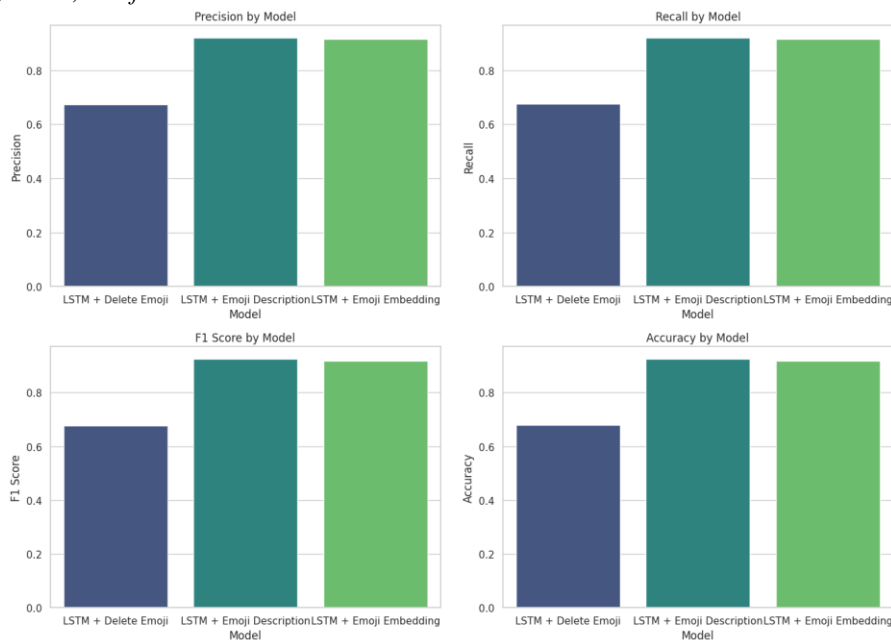


Figure 2. Evaluation Matriks

Based on the value of the evaluation matrix, the *LSTM + Delete Emoji* model is lower than the *LSTM + Emoji Description* and *LSTM + Emoji Embedding* models, because when clacifying, emoji is very influential. When deleting emoji, it will result in the loss of important information that affects the meaning of the sentence. And the *LSTM + Emoji Description* model produces better, because using emoji descriptions and descriptions from text will retain important information in the sentence, so the model will be better at capturing the meaning of emojis. And in *LSTM + Emoji Embedding* is also better because by changing emoji with vector representation the model can learn the semantic relationship between emoji and words in the text, thus maintaining information from emoji.

4. CONCLUSION

Based on the experiments conducted, it can be concluded that the use of emoji has a significant role in improving the accuracy of *hate speech* classification which is highly dependent on the presence of emoji as a label determinant. There are three models implemented in this experiment, namely *LSTM + Emoji Embedding*, *LSTM + Emoji Description*, and *LSTM + Delete Emoji*. The experimental results show that the model that uses emoji (both in the form of embedding and description) has a much higher accuracy than the model that deletes emoji. The *LSTM + Emoji Description* model achieved an accuracy of 0.92, and the *LSTM + Emoji Embedding* model achieved an accuracy of 0.918. In contrast, the *LSTM + Delete Emoji* model only achieved an accuracy of 0.68. Therefore, in *hate speech* classification tasks involving emoji, it is highly

recommended to consider the use of emoji both in the form of direct embedding and description to obtain optimal results. In this research, there are several limitations that need to be considered. One of the limitations is the inadequate representation of the dataset. A lack of representation in the dataset can cause the resulting model to be unable to generalize well to new data, leading to overfitting. Therefore, augmentation was carried out to enrich the information in the dataset, enabling the model to better generalize new data. For future research, several suggestions need to be considered to improve the quality of research results. First, the use of synonym augmentation techniques can be applied to enrich data variations without changing the original meaning of the text. Additionally, searching for additional data from relevant sources can be undertaken to enhance the dataset's information without solely relying on augmentation techniques. Finally, the application of transfer learning can be explored as an alternative to utilizing pre-trained models with better generalization capabilities, thereby improving overall model accuracy and performance.

REFERENCES




- [1] H. A. Le, T. Tao, P. Dinh, and N. T. Nguyen, "Advances in Intelligent Systems and Computing 360 Modelling, Computation and Optimization in Information Systems and Management Sciences Proceedings of the 3rd International Conference on Modelling, Computation and Optimization in Information Systems and Management Sciences-MCO 2015-Part II." [Online]. Available: <http://www.springer.com/series/11156>
- [2] M. O. Ibrohim and I. Budi, "Hate speech and abusive language detection in Indonesian social media: Progress and challenges," Aug. 01, 2023, *Elsevier Ltd.* doi: 10.1016/j.heliyon.2023.e18647.
- [3] A. F. Adiyaksa, D. Richasdy, and A. F. Ihsan, "Hate Speech Detection on YouTube Using Long Short-Term Memory and Latent Dirichlet Allocation Method," *Journal of Information System Research (JOSH)*, vol. 3, no. 4, pp. 644–650, Jul. 2022, doi: 10.47065/josh.v3i4.1875.
- [4] C. N. Arbaatun, D. Nurjanah, and H. Nurrahmi, "Hate Speech Detection on Twitter through Natural Language Processing using LSTM Model," *Building of Informatics, Technology and Science (BITS)*, vol. 4, no. 3, 2022, doi: 10.47065/bits.v4i3.2718.
- [5] A. Singh, E. Blanco, and W. Jin, "Incorporating Emoji Descriptions Improves Tweet Classification," Association for Computational Linguistics. [Online]. Available: <https://github.com/uclmr/>
- [6] S. García, J. Luengo, and F. Herrera, "Intelligent Systems Reference Library 72 Data Preprocessing in Data Mining." [Online]. Available: <http://www.springer.com/series/8578>
- [7] Z. Zhang and L. Luo, "Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter," Feb. 2018, [Online]. Available: <http://arxiv.org/abs/1803.03662>
- [8] S. Ghosal, A. Jain, D. K. Tayal, V. G. Menon, and A. Kumar, "Inculcating Context for Emoji Powered Bengali Hate Speech Detection using Extended Fuzzy SVM and Text Embedding Models," *ACM Transactions on Asian and Low-Resource Language Information Processing*, Mar. 2023, doi: 10.1145/3589001.
- [9] E. Barry, S. Jameel, and H. Raza, "Emojional: Emoji Embeddings." [Online]. Available: <https://unicode.org/emoji/charts/full-emoji-list.html>
- [10] A. M. Peco Chacón, I. Segovia Ramírez, and F. P. García Márquez, "K-nearest neighbour and K-fold cross-validation used in wind turbines for false alarm detection," *Sustainable Futures*, vol. 6, Dec. 2023, doi: 10.1016/j.sfr.2023.100132.
- [11] M. Lukauskas, V. Pilinkienė, J. Bruneckienė, A. Stundžienė, A. Grybauskas, and T. Ruzgas, "Economic Activity Forecasting Based on the Sentiment Analysis of News," *Mathematics*, vol. 10, no. 19, Oct. 2022, doi: 10.3390/math10193461.
- [12] H. R. Kirk, B. Vidgen, P. Röttger, T. Thrush, and S. A. Hale, "HATEMOJI: A Test Suite and Adversarially-Generated Dataset for Benchmarking and Detecting Emoji-Based Hate."
- [13] M. A. Rosid, A. S. Fitriani, I. R. I. Astutik, N. I. Mulloh, and H. A. Gozali, "Improving Text Preprocessing for Student Complaint Document Classification Using Sastrawi," in *IOP Conference Series: Materials Science and Engineering*, Institute of Physics Publishing, Jul. 2020. doi: 10.1088/1757-899X/874/1/012017.
- [14] F. Aftab *et al.*, "A Comprehensive Survey on Sentiment Analysis Techniques," *International Journal of Technology*, vol. 14, no. 6, pp. 1288–1298, 2023, doi: 10.14716/ijtech.v14i6.6632.
- [15] D. Haryadi and G. Putra Kusuma, "Emotion Detection in Text using Nested Long Short-Term Memory," 2019. [Online]. Available: www.ijacsa.thesai.org
- [16] Z. Zhang and L. Luo, "Semantic Web 1 (0) 1-5 1 IOS Press."
- [17] A. R. Dainas and S. C. Herring, "Interpreting Emoji Pragmatics."
- [18] R. Hataya, J. Zdenek, K. Yoshizoe, and H. Nakayama, "Faster AutoAugment: Learning Augmentation Strategies using Backpropagation," Nov. 2019, [Online]. Available: <http://arxiv.org/abs/1911.06987>

- [19] S. Gupta, P. Priyadarshi, and M. Gupta, "Hateful Comment Detection and Hate Target-Type Prediction for Video Comments," in *International Conference on Information and Knowledge Management, Proceedings*, Association for Computing Machinery, Oct. 2023, pp. 3923–3927. doi: 10.1145/3583780.3615260.
- [20] S. Y. Feng *et al.*, "A Survey of Data Augmentation Approaches for NLP." [Online]. Available: <https://github.com/styfeng/DataAug4NLP>.
- [21] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, "Image Data Augmentation for Deep Learning: A Survey."
- [22] A. Kurniasih and L. P. Manik, "On the Role of Text Preprocessing in BERT Embedding-based DNNs for Classifying Informal Texts," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 6, pp. 927–934, 2022, doi: 10.14569/IJACSA.2022.01306109.
- [23] W. Bourequat and H. Mourad, "Sentiment Analysis Approach for Analyzing iPhone Release using Support Vector Machine," *International Journal of Advances in Data and Information Systems*, vol. 2, no. 1, pp. 36–44, Apr. 2021, doi: 10.25008/ijadis.v2i1.1216.
- [24] N. Babanejad, A. Agrawal, A. An, and M. Papagelis, "A Comprehensive Analysis of Preprocessing for Word Representation Learning in Affective Tasks." [Online]. Available: <https://github.com/NastaranBa/>
- [25] D. Kent and F. Salem, "Performance of Three Slim Variants of The Long Short-Term Memory (LSTM) Layer."

BIOGRAPHIES OF AUTHORS (10 PT)

The recommended number of authors is at least 2. One of them as a corresponding author.

Please attach clear photo (3x4 cm) and vita. Example of biographies of authors:

	Yoga Sihombing is a student at the Department of Information Systems, Del Institute of Technology, Indonesia. He can be contacted via email: yogasihombing2001@gmail.com
	Gabriel Saputra Panggabean is a student at the Department of Information Systems, Del Institute of Technology, Indonesia. Gabriel Saputra Panggabean has experience in web development and mobile application design systems besides machine learning algorithms. He can be contacted via email: gabrielpanggabean1212@gmail.com .
	Mares G H Siagian is a student at the Department of Information Systems, Del Institute of Technology, Indonesia. Mares G H Siagian has experience in web development and mobile application design systems besides machine learning algorithms. He can be contacted via email: maressiagian@gmail.com .