Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control

Journal homepage: http://kinetik.umm.ac.id

ISSN: 2503-2267

Vol. 4, No. 3, August 2019, Pp. 197-206

Effect of Stopword Removal on Sentiment Analysis using LSTM in Indonesian Language

Fahrendra Khoirul Ihtada*[1], Mohammad Yoga Pratama[2], Khadijah Fahmi Hayati Holle[3]
1, Informatic Engineering, Universitas Islam Negeri Maulana Malik Ibrahim, Indonesia
2, Informatic Engineering, Universitas Islam Negeri Maulana Malik Ibrahim, Indonesia
3, Informatic Engineering, Universitas Islam Negeri Maulana Malik Ibrahim, Indonesia

Article Info

Keywords:
Sentiment Analysis, Indonesian Language, LSTM, Preprocessing, Stopword Removal, Model Performance, Informative Context.

Article history:
Received 17 August 2018
Revised 15 February 2019
Accepted 4 April 2019
Available online 4 April 2019

Cite:
Wardana, A., Rakhmatsyah, A., Minarno, A., & Anbiya, D. (2019). Internet of Things Platform for Manage Multiple Message Queuing Telemetry Transport Broker Server. Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 4(3). doi:http://dx.doi.org/10.22219/kinetik.v4i3.841

* Corresponding author.
Corresponding Author
E-mail address:
author@email.ac.id

Abstract

This research investigates sentiment analysis of Indonesian social media comments using the SmSA (IndoNLU) dataset. The study focuses on analyzing the sentiments - positive, neutral and negative - expressed by users in response to various topics discussed online. Long Short-Term Memory Networks (LSTM) are used for their ability to capture sequential contextual relationships in textual data. The study aims to provide useful insights for companies to improve their products and services based on user perceptions and feedback. The research methodology includes data collection, pre-processing, sentiment classification using LSTM and evaluation using Confusion Matrix and Accuracy metrics. The results show the impact of different preprocessing techniques, such as stopword removal, on model performance. The no stop model consistently better than the stop model, indicating the importance of informative context in sentiment classification. The research concludes by suggesting further exploration of alternative preprocessing approaches and larger datasets to improve the accuracy and generalizability of sentiment analysis models in the Indonesian language.

1. Introduction

In today's ever-evolving world, sentiment analysis is one of the most frequently discussed topics in the field of natural language processing. Many social media platforms such as Instagram, Twitter, and Facebook provide a place for users to share opinions, comments, and feedback on various topics. Therefore, analyzing sentiment in text is important to understand people's opinions and responses to a topic that is being discussed[1].

This research aims to analyze sentiment in comments and responses in Indonesian on the SmSA (IndoNLU) dataset using the Long Short-Term Memory Networks (LSTM) method[2]. We want to know users' responses and comments on a topic expressed through social media platforms in the dataset. By analyzing, we can identify sentiment trends that are being discussed.

The dataset we use is available on the internet, the SmSA dataset (IndoNLU). This dataset consists of 12,760 rows of data with Indonesian language that has been categorized into three labels: positive, neutral, and negative. This dataset is the result of collecting user comments and responses from various online platforms. The diversity of comments already in the dataset allows for a more comprehensive sentiment analysis[3].

This research uses the Long Short-Term Memory Networks (LSTM) method. This method is part of a Recurrent Neural Network (RNN) architecture that is good at modeling contextual relationships in sequential data, such as text in comments and responses. This method has been widely used and successful in sentiment analysis in various languages[4].

It is expected that sentiment analysis of the SmSA (IndoNLU) dataset used in this study to gain insight into how users react to a topic, product, or service shared through an online platform. the implications of this research can help a company or others improve the service or quality of the products they release and understand how users perceive them.

2. Research Method

For the research method, we briefly describe the experimental design. we went through several stages. The first step is data collection, which we get from public dataset. after that, preprocessing must be done. this will standardise text data, which will reduce the dimensionality of the feature space and eliminate redundancies. so improve classification performance and provide more meaningful insights from sentiment analysis tasks.The last process is sentiment classification using Recurrent Neural Network which is LSTM based. The result is evaluate the performance with 2 metrics, Confusion Matrix and Accuracy.

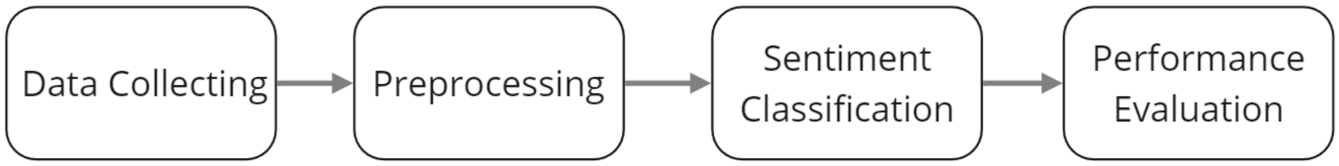


Figure 1 : Flow System

2.1. Dataset

The dataset used in this study is the SmSA (IndoNLU) dataset [3], which is a publicly available dataset. It consists of 12,760 data rows categorized into three sentiment labels: positive, negative and neutral. Within the dataset, there are 7,358 rows of data labeled as positive, 1,366 rows of data labeled as neutral and 4,036 rows of data labeled as negative. This dataset provides a valuable resource for sentiment analysis tasks, allowing researchers and practitioners to analyze and classify text data based on sentiment polarity.

Table 1. Detail Dataset

Sentiment	Quantity
Negative	4034
Neutral	1366
Positive	7358

2.2. Preprocessing

Data preprocessing is an important step in data analysis, as it involves cleaning the raw data and transforming it into a format suitable for further analysis. In sentiment analysis for text classification, preprocessing techniques play an important role in improving the accuracy and efficiency of classification algorithms. The main purpose of preprocessing is to improve the quality of the input data by removing noise, irrelevant information and inconsistencies. This allows for more effective analysis and interpretation of the data. Preprocessing can standardize text data, reduce the dimensionality of the feature space and eliminate redundancies. Ultimately, these steps contribute to improved classification performance and provide more meaningful insights from sentiment analysis tasks. The pre-processing techniques that have been used in this study are as follows :

a. Case folding: This step converts all the letters in a text document to lowercase. It helps to standardize text and avoid inconsistencies caused by different capitalisation [5] .
b. Stemming: The purpose of stemming is to convert words to their root form by removing prefixes and suffixes. In this study, the Sastrawi Stemmer is used to perform stemming[6], which helps to improve the efficiency of the analysis.
c. Stopword Removal: Stopwords are frequently occurring words that have no significant meaning in a text. Examples of Indonesian stopwords are 'yang', 'di', 'untuk' and 'dari'. The study uses the Sastrawi stopword list to remove these stopwords[7], thereby removing noise and reducing the dimensionality of the data.
d. Tokenization: Tokenization is the process of breaking down a full text string into a list of individual words or tokens[8]. It helps to focus the analysis on the meaningful units of the text, making further processing and analysis easier.
e. Categorical label encoding: The TensorFlow Keras library is used to encode the sentiment labels as categorical variables in order to prepare them for classification. This encoding process ensures that the labels are represented in a format suitable for the classification task.

By implementing these preprocessing steps, the study aims to improve the quality of the data, reduce noise, and improve the accuracy and efficiency of sentiment analysis for text classification.

Table 2. Preprocessing Model

Preprocessing Model	Preprocessing
Stop	With stopwords removal
No Stop	Without stopwords removal

2.3. Sentiment Classification

The sentiment classification algorithm is based on a neural network, specifically LSTM. LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) architecture commonly used for sequence modeling and processing tasks, especially in the field of natural language processing (NLP)[9]. It is designed to overcome the vanishing gradient problem that occurs in traditional RNNs, which struggle to capture long-term dependencies in sequential data. Here is the model we've proposed for this study and our system architecture[10].

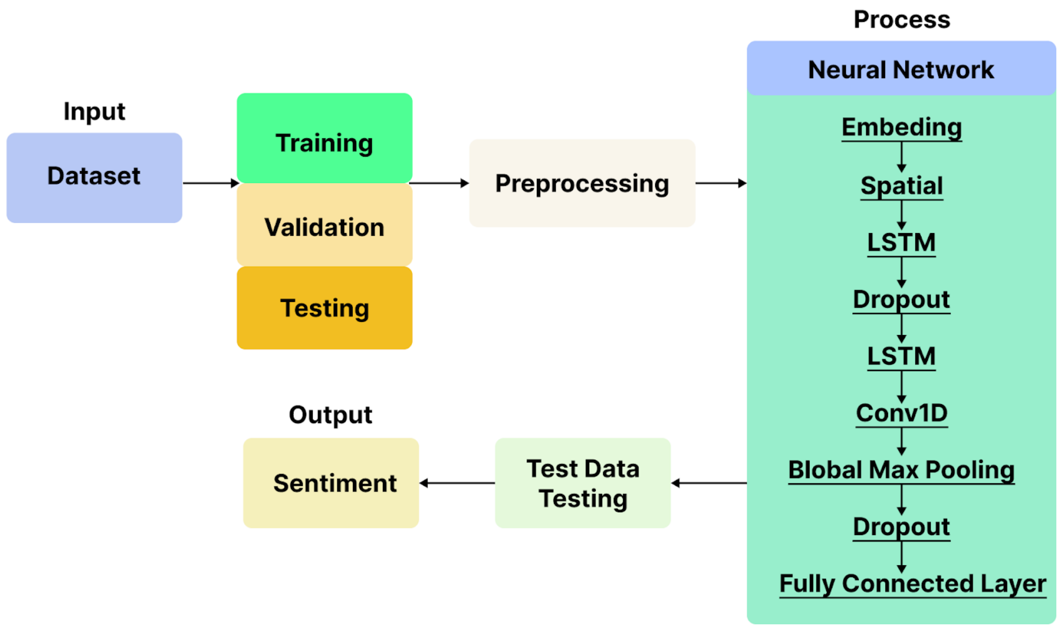


Figure 2 : System Design

Table 3. Mode Architecturel

Type	Output Shape	Node
Embedding	None, 100, 32	
SpatialDropout1D	None, 100, 32	
LSTM	None, 100, 128	82432
Dropout	None, 100, 128	
LSTM	None, 100, 64	49408
Conv1D	None, 96, 64	20544
GlobalMaxPooling1D	None, 64	
Dropout	None, 64	
Dense	None, 32	2080
Dense_1	None, 3	99

The proposed model for this study consists of several layers:

1. Embedding Layer: This layer converts words represented as numbers into fixed-size dense vectors known as embeddings. By learning the context-based representations of words, the model can capture the meaning and relationships between words.
2. SpatialDropout1D Layer: Dropout is a technique used to prevent overfitting by randomly ignoring input features during training. In this case, the SpatialDropout1D layer applies dropout specifically to the one-dimensional input, which is the sequence of embedded word vectors. By randomly dropping some vectors, the model becomes more robust and can generalize better to unseen data.
3. LSTM Layer: The LSTM layer is a key component of the model. It processes input sequences and maintains an internal state memory, enabling it to capture long-term dependencies and understand the context of words within the sequence.
4. Dropout Layer: This dropout layer applies regularization to the outputs of the previous LSTM layer. By randomly setting a fraction of input units to 0 during training, it helps prevent the model from relying too heavily on specific features and overfitting the data.
5. LSTM Layer: Similar to the previous LSTM layer, this layer also processes input sequences and maintains an internal state memory[7]. However, it has fewer units compared to the previous LSTM layer, which helps simplify the model and prevent overfitting.
6. Conv1D Layer: The Conv1D layer performs one-dimensional convolution on the input sequence. It uses a sliding window to capture local patterns or features in the text data.
7. GlobalMaxPooling1D Layer: After the convolutional layer, the GlobalMaxPooling1D layer reduces the dimensionality of the features while retaining the most important information. It selects the maximum value from each feature map, creating a fixed-length representation of the sequence.
8. Dropout Layer: This dropout layer applies regularization to the outputs of the previous layer, further preventing overfitting and enhancing the model's robustness.
9. Dense Layer: The dense layer is a fully connected layer that learns non-linear transformations of the input data. It extracts higher-level features from the input text using the ReLU activation function.
10. Dense Layer: This is the output layer of the model, with 3 units corresponding to the three sentiment labels: negative, neutral, and positive. The dense layer applies the softmax activation function to produce probability scores for each class, indicating the predicted sentiment of the input text.

These layers collectively process the input text data, capture relevant features, and make predictions about the sentiment.

2.4. Performance Evaluation

Performance results are measured using the following metrics[11]: Confusion Matrix that include Precision, Recall, F1 Score and Accuracy.

$$Recall = \frac{TP}{TP + FN}$$

Recall (also called sensitivity): Recall tells us how well the model correctly identifies positive cases. It shows the proportion of actual positive cases that the model predicts to be positive. A high recall means that the model is good at finding positive cases and avoiding false negatives.

$$Precision = \frac{TP}{TP+FP}$$

Precision: Precision tells us how accurate the model is at predicting positive cases[12]. It measures the proportion of correctly predicted positive cases out of all cases predicted to be positive. A high precision means that the model has fewer false positives, indicating that it is reliable in identifying positive cases.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

F1 Score: The F1 score combines both recall and precision into a single metric[13]. It provides a balanced measure by taking into account both false positives and false negatives. The F1 score is helpful when we want to find a good balance between precision and recall.

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN}$$

Accuracy: Accuracy measures how well the model performs overall by calculating the proportion of correctly predicted instances out of all instances[14]. It provides a general assessment of the model's accuracy. However, accuracy may not be the best metric if the classes are unbalanced, as it can be misleading.

These metrics help to understand the strengths and weaknesses of classification models. Recall and precision focus on specific aspects of the model's predictions, while the F1 score provides a balanced view. Accuracy gives an overall picture of the model's performance, but may not be appropriate if the classes are unbalanced.

3. Results and Discussion

This study is implemented using the Python programming language, using popular machine learning libraries such as scikit-learn and TensorFlow.In this experiment, we use 1 algorithm and we will compare between datasets that apply stopword removal and those that do not apply stopword removal[15]. Initially, the models are trained for 100 epochs, which represent complete passes through the training data. However, in order to monitor and improve the performance of the model[16], a callback mechanism is used. This callback is activated when the validation accuracy of the model shows no improvement for several consecutive epochs of training[17]. By implementing this callback, model training can be stopped early if no significant progress is observed, saving computational resources and preventing overfitting.

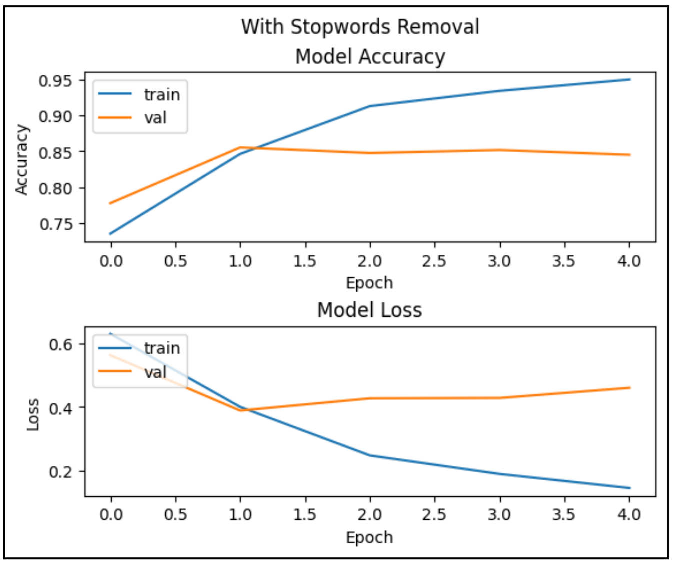
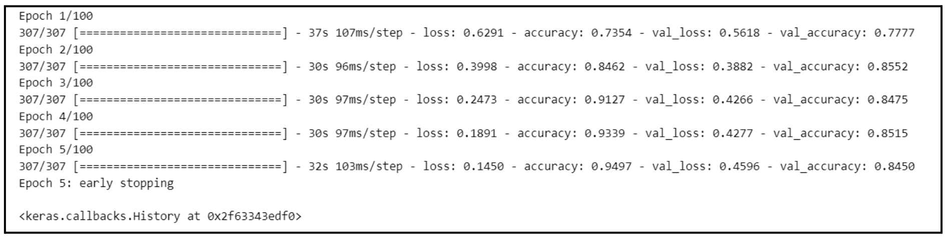


Figure 3. Model Performance with Stopwords Removal



```
Epoch 1/100
307/307 [==============================] - 37s 107ms/step - loss: 0.6291 - accuracy: 0.7354 - val_loss: 0.5618 - val_accuracy: 0.7777
Epoch 2/100
307/307 [==============================] - 30s 96ms/step - loss: 0.3998 - accuracy: 0.8462 - val_loss: 0.3882 - val_accuracy: 0.8552
Epoch 3/100
307/307 [==============================] - 30s 97ms/step - loss: 0.2473 - accuracy: 0.9127 - val_loss: 0.4266 - val_accuracy: 0.8475
Epoch 4/100
307/307 [==============================] - 30s 97ms/step - loss: 0.1891 - accuracy: 0.9339 - val_loss: 0.4277 - val_accuracy: 0.8515
Epoch 5/100
307/307 [==============================] - 32s 103ms/step - loss: 0.1450 - accuracy: 0.9497 - val_loss: 0.4596 - val_accuracy: 0.8450
Epoch 5: early stopping

<keras.callbacks.History at 0x2f63343edf0>
```

Figure 4. Model Performance with Stopwords Removal

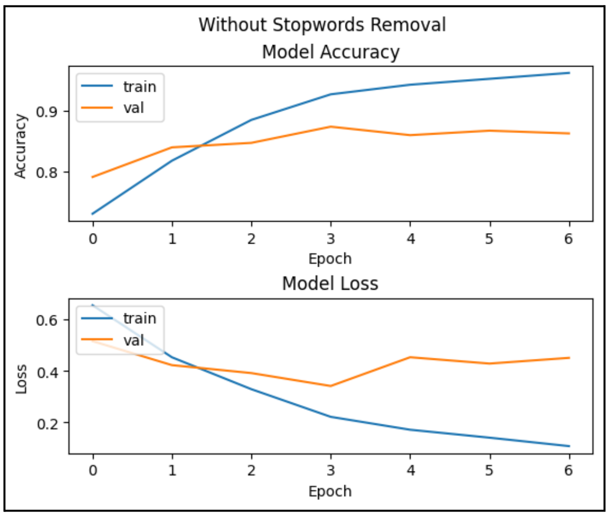
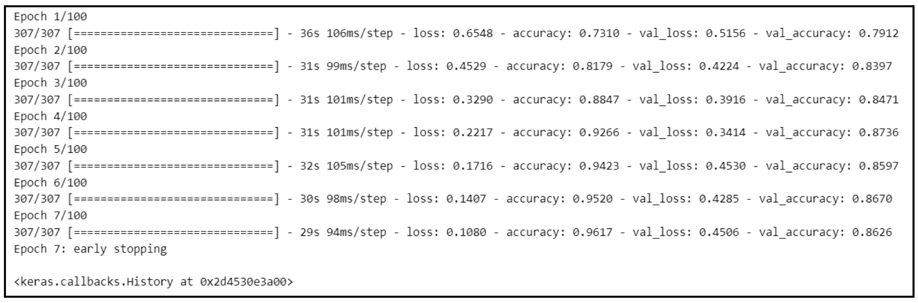


Figure 5. Model Performance without Stopwords Removal



```
Epoch 1/100
307/307 [==============================] - 36s 106ms/step - loss: 0.6548 - accuracy: 0.7310 - val_loss: 0.5156 - val_accuracy: 0.7912
Epoch 2/100
307/307 [==============================] - 31s 99ms/step - loss: 0.4529 - accuracy: 0.8179 - val_loss: 0.4224 - val_accuracy: 0.8397
Epoch 3/100
307/307 [==============================] - 31s 101ms/step - loss: 0.3290 - accuracy: 0.8847 - val_loss: 0.3916 - val_accuracy: 0.8471
Epoch 4/100
307/307 [==============================] - 31s 101ms/step - loss: 0.2217 - accuracy: 0.9266 - val_loss: 0.3414 - val_accuracy: 0.8736
Epoch 5/100
307/307 [==============================] - 32s 105ms/step - loss: 0.1716 - accuracy: 0.9423 - val_loss: 0.4530 - val_accuracy: 0.8597
Epoch 6/100
307/307 [==============================] - 30s 98ms/step - loss: 0.1407 - accuracy: 0.9520 - val_loss: 0.4285 - val_accuracy: 0.8670
Epoch 7/100
307/307 [==============================] - 29s 94ms/step - loss: 0.1080 - accuracy: 0.9617 - val_loss: 0.4506 - val_accuracy: 0.8626
Epoch 7: early stopping

<keras.callbacks.History at 0x2d4530e3a00>
```

Figure 6. Training Process without Stopwords Removal

Table 4. Model Performance in Training

Preprocessing Model	Preprocessing	Accuracy	Loss
Stop	With stopwords removal	**0.9497**	0.1450
No Stop	Without stopwords removal	0.9617	**0.1**

Two different models were used in the training process, one with stop word removal (Stop Model) and one without (No Stop Model). The Stop model achieved a recall at epoch 5, with an accuracy of 0.9497 and a loss of 0.1450. In contrast, the No Stop model achieved a callback at epoch 7 with a higher accuracy of 0.9617 and a lower loss of 0.1.

These results indicate a reasonable performance from both models in terms of accuracy, with the No Stop model showing a little better performance than the Stop model[18]. The use of stop word removal in the preprocessing step may have resulted in a little less accuracy for the Stop model. The difference in loss values also indicates that the No Stop model showed a better optimization during training[8], as indicated by its lower loss[19]. This difference could be attributed to the informative context provided by the stop words, which helped the No Stop model to effectively capture sentiment patterns.

Table 5. Result Each Model on Testing Data

Preprocessing Model	Sentiment	Precision	Recall	F1 Score
Stop	Negative	0.80	0.79	0.80
	Neutral	0.74	0.68	0.71
	Positive	0.88	0.91	0.89
No Stop	Negative	0.81	0.84	0.86
	Neutral	0.78	0.74	0.76
	Positive	0.91	0.90	0.90

Table 6. Accuracy Each Model on Testing Data

Preprocessing Model	Accuracy
Stop	0.85
No Stop	**0.86**

Both models were then tested on the test data to evaluate their performance. The Stop model showed the following results: a precision of 0.80 for negative sentiment, a recall of 0.79 and an F1 score of 0.80. Neutral sentiment had a precision of 0.74, a recall of 0.68 and an F1 score of 0.71. Finally, positive sentiment had a precision of 0.88, a recall of 0.91 and an F1 score of 0.89.

Similarly, the No Stop model showed the following results: a precision of 0.81 for negative sentiment, a recall of 0.84 and an F1 score of 0.83. For neutral sentiment, the precision was 0.78, the recall was 0.74 and the F1 score was 0.76. For positive sentiment, a precision of 0.91, a recall of 0.90 and an F1 score of 0.90 were achieved.

The evaluation results highlight the performance of each model on different sentiment classes. The Stop model showed relatively lower scores for neutral sentiment compared to the No Stop model[20], which indicates that the use of stop words in the pre-processing step of the Stop model may have affected its ability to accurately classify neutral sentiment. On the other hand, the No Stop Model showed consistent and good performance across all sentiment classes, with higher precision, recall and F1 scores.

4. Conclusion

In conclusion, this study tried to compare the performance of sentiment analysis models using different preprocessing techniques in the context of the Indonesian language. Two models were used: one with stop word removal (Stop Model) and one without (No Stop Model). The results showed that both models achieved reasonable accuracy, with the No Stop model performing a little better. On test data, the Stop model achieved 85% accuracy and the No Stop model achieved 86% accuracy. This indicates that the use of stop words in the preprocessing step may have affected the accuracy of the stop model.

Also, evaluation on the test data showed that the No Stop model consistently beat the Stop model across all sentiment classes. It had higher precision, recall and F1 scores, indicating its effectiveness in capturing sentiment patterns in Indonesian text. However, the Stop Model faced challenges in correctly classifying neutral sentiment, possibly because of the removal of informative context provided by stop words.

Further research can explore alternative preprocessing approaches and experiment with larger datasets to improve the overall accuracy and generalizability of sentiment analysis models in the Indonesian language.

Acknowledgement

Sincere thanks to Universitas Islam Negeri Maulana Malik Ibrahim for the support and facilities that have been provided during this research. We also thank the SmSA dataset provider (IndoNLU) for providing the dataset for free and freely accessible during this research. We really appreciate their efforts in collecting and labeling the dataset so that we get good resources in conducting this research.

Without the support of Maulana Malik Ibrahim State Islamic University and the provision of the SmSA (IndoNLU) dataset, this research would not have been able to proceed properly.

References

[1] A. A. Firdaus, A. Yudhana, and I. Riadi, “Public opinion analysis of presidential candidate using naïve bayes method,” vol. 4, no. 2, 2023.

[2] S. Cahyaningtyas, D. Hatta Fudholi, and A. Fathan Hidayatullah, “Deep Learning for Aspect-Based Sentiment Analysis on Indonesian Hotels Reviews,” *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 4, no. 3, 2021, doi: 10.22219/kinetik.v6i3.1300.

[3] B. Wilie *et al.*, “IndoNLU: Benchmark and Resources for Evaluating Indonesian Natural Language Understanding,” 2020, [Online]. Available: http://arxiv.org/abs/2009.05387

[4] R. Majid and H. A. Santoso, "Conversations Sentiment and Intent Categorization Using Context RNN for Emotion Recognition," *2021 7th Int. Conf. Adv. Comput. Commun. Syst. ICACCS 2021*, pp. 46–50, 2021, doi: 10.1109/ICACCS51430.2021.9441740.

[5] Z. Zong and C. Hong, "On application of natural language processing in machine translation," *Proc. - 2018 3rd Int. Conf. Mech. Control Comput. Eng. ICMCCE 2018*, pp. 506–510, 2018, doi: 10.1109/ICMCCE.2018.00112.

[6] H. A. Almuzaini and A. M. Azmi, "Impact of Stemming and Word Embedding on Deep Learning-Based Arabic Text Categorization," *IEEE Access*, vol. 8, pp. 127913–127928, 2020, doi: 10.1109/ACCESS.2020.3009217.

[7] A. W. Pradana and M. Hayaty, "The Effect of Stemming and Removal of Stopwords on the Accuracy of Sentiment Analysis on Indonesian-language Texts," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 4, no. 3, pp. 375–380, 2019, doi: 10.22219/kinetik.v4i4.912.

[8] N. Lee, K. Kim, and T. Yoon, "Implementation of robot journalism by programming custombot using tokenization and custom tagging," *Int. Conf. Adv. Commun. Technol. ICACT*, pp. 566–570, 2017, doi: 10.23919/ICACT.2017.7890154.

[9] M. R. Haque, S. Akter Lima, and S. Z. Mishu, "Performance Analysis of Different Neural Networks for Sentiment Analysis on IMDb Movie Reviews," *3rd Int. Conf. Electr. Comput. Telecommun. Eng. ICECTE 2019*, pp. 161–164, 2019, doi: 10.1109/ICECTE48615.2019.9303573.

[10] H. C. Tissot *et al.*, "Natural Language Processing for Mimicking Clinical Trial Recruitment in Critical Care: A Semi-Automated Simulation Based on the LeoPARDS Trial," *IEEE J. Biomed. Heal. Informatics*, vol. 24, no. 10, pp. 2950–2959, 2020, doi: 10.1109/JBHI.2020.2977925.

[11] N. Hossain, M. R. Bhuiyan, Z. N. Tumpa, and S. A. Hossain, "Sentiment Analysis of Restaurant Reviews using Combined CNN-LSTM," *2020 11th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2020*, 2020, doi: 10.1109/ICCCNT49239.2020.9225328.

[12] O. F. Prihono and P. K. Sari, "Comparison Analysis Of Social Influence Marketing For Mobile Payment Using Support Vector Machine," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 4, no. 3, pp. 367–374, 2019, doi: 10.22219/kinetik.v4i4.921.

[13] S. Cao and P. Gao, "LSTM-Gate CNN network for Aspect Sentiment Analysis," *Proc. - 2020 5th Int. Conf. Inf. Sci. Comput. Technol. Transp. ISCTT 2020*, pp. 443–447, 2020, doi: 10.1109/ISCTT51595.2020.00084.

[14] J. Gao, R. Yao, H. Lai, and T. C. Chang, "Sentiment Analysis with CNNs Built on LSTM on Tourists Comments," *Proc. 2019 IEEE Eurasia Conf. Biomed. Eng. Healthc. Sustain. ECBIOS 2019*, no. 2, pp. 108–111, 2019, doi: 10.1109/ECBIOS.2019.8807844.

[15] S. Gharatkar, A. Ingle, T. Naik, and A. Save, "Review preprocessing using data cleaning and stemming technique," *Proc. 2017 Int. Conf. Innov. Information, Embed. Commun. Syst. ICIIECS 2017*, vol. 2018-Janua, pp. 1–4, 2018, doi: 10.1109/ICIIECS.2017.8276011.

[16] F. J. Ariza-López, J. Rodríguez-Avi, and M. V. Alba-Fernández, "Complete control of an observed confusion matrix," *Int. Geosci. Remote Sens. Symp.*, vol. 2018-July, pp. 1222–1225, 2018, doi: 10.1109/IGARSS.2018.8517540.

[17] M. Karrabi, L. Oskooie, M. Bakhtiar, M. Farahani, and R. Monsefi, "Sentiment Analysis of Informal Persian Texts Using Embedding Informal words and Attention-Based LSTM Network," *8th Iran. Jt. Congr. Fuzzy Intell. Syst. CFIS 2020*, pp. 143–147, 2020, doi: 10.1109/CFIS49607.2020.9238699.

[18] F. Alzami, E. D. Udayanti, D. P. Prabowo, and R. A. Megantara, "Document Preprocessing with TF-IDF to Improve the Polarity Classification Performance of Unstructured Sentiment Analysis," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 4, no. 3, pp. 235–242, 2020, doi: 10.22219/kinetik.v5i3.1066.

[19] B. Kholifah, I. Syarif, and T. Badriyah, "Mental Disorder Detection via Social Media Mining using Deep Learning," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 4, pp. 309–316, 2020, doi: 10.22219/kinetik.v5i4.1120.

[20] W. Yue and L. Li, "Sentiment Analysis using Word2vec-CNN-BiLSTM Classification," pp. 3–7.