IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.14, No.2, April 2020, pp. 169~178
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: 10.22146/ijccs.51743

Bidirectional Long Short Term Memory Method and Word2vec Extraction Approach for Hate Speech Detection

Auliya Rahman Isnain*[1], Agus Sihabuddin[2], Yohanes Suyanto[3]
[1]Master Program of Computer Science, FMIPA UGM, Yogyakarta, Indonesia
[2,3]Department of Computer Science and Electronics, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1]rahman.isnain@gmail.com**, [2]a_sihabudin@ugm.ac.id, [3]yanto@ugm.ac.id

Abstrak

Saat ini pembahasan mengenai ujaran kebencian di Indonesia sedang hangat, terutama melalui media sosial. Ujaran kebencian merupakan komunikasi yang meremehkan sesorang atau kelompok berdasarkan karakteristik seperti (ras, etnis, jenis kelamin, kewarganegaraan, agama dan oragnisasi). Twitter salah satu media sosial yang digunakan seseorang untuk mengutarakan perasaan dan opini melalui tweet, termasuk tweet yang megandung ujaran kebencian. Karena twitter mempunyai pengaruh besar bagi kesuksesan ataupun kehancuran citra seseorang.

Penelitian ini bertujuan untuk mendeteksi ujaran kebencian atau bukan ujaran kebencian tweet berbahasa Indonesiadengan mengunakan metode Bidirectional Long Short Term Memory dan metode ekstrasi fitur word2vec dengan arsitektur Continuous bag-of-word (CBOW). Untuk pengujian metode BiLSTM dengan perhitungan nilai akurasi, presisi, recall, dan F-measure.

Penggunaan word2vec dan metode Bidirectional Long Short Term Memory dengan arsitektur CBOW, dengan epoch 10, learning rate 0.001 dan jumlah neuron 200 pada layer tersembunyi, menghasilkan tingkat akurasi 94,66%, dengan masing-masing nilai presisi 99,08%, recall 93,74% dan F-measure 96,29%. Sedangkan untuk Bidirectional Long Short Term Memory dengan tiga layer memiliki akurasi 96,93%. Penambahan satu layer pada BiLSTM meningkat 2,27%.

Kata kunci— *Ujaran Kebencian, LSTM, BiLSTM, Word2vec, CBOW, Skipgram, Twitter*

Abstract

Currently, the discussion about hate speech in Indonesia is warm, primarily through social media. Hate speech is communication that disparages a person or group based on characteristics such as (race, ethnicity, gender, citizenship, religion and organization). Twitter is one of the social media that someone uses to express their feelings and opinions through tweets, including tweets that contain expressions of hatred because Twitter has a significant influence on the success or destruction of one's image.

This study aims to detect hate speech or not hate Indonesian speech tweets by using the Bidirectional Long Short Term Memory method and the word2vec feature extraction method with Continuous bag-of-word (CBOW) architecture. For testing the BiLSTM purpose with the calculation of the value of accuracy, precision, recall, and F-measure.

The use of word2vec and the Bidirectional Long Short Term Memory method with CBOW architecture, with epoch 10, learning rate 0.001 and the number of neurons 200 on the hidden layer, produce an accuracy rate of 94.66%, with each precision value of 99.08%, recall 93, 74% and F-measure 96.29%. In contrast, the Bidirectional Long Short Term Memory with three layers has an accuracy of 96.93%. The addition of one layer to BiLSTM increased by 2.27%.

Keywords— *Hate Speech, LSTM, BiLSTM, Word2vec, CBOW, Skipgram, Twitter*

Received November 21th,2019; Revised April 10th, 2020; Accepted April 29th, 2020

1. INTRODUCTION

Social media uses web-based internet technology that converts communication data into conversations between users of social media [1] . Twitter is a social media that allows users to express feelings and opinions through tweets, including tweets that contain expressions of hatred [2]. Twitter has a significant influence on the success or destruction of one's image [3].

Hate speech or hate speech is a communication that belittles a person or group based on characteristics such as (race, ethnicity, gender, nationality, religion and organization) [4]. Expressing hate speech has become a trend, and many people use this as a shortcut to gain instant popularity without putting more effort [5]. Dissemination of information containing hate speech can spread quickly and widely. According to investigators from the Directorate of Criminal Acts Siber Bareskrim, said the majority of cybercrime is dominated by defamation and hate speech on social media with a percentage of 80% [6].

Research on the detection of hate speech for Indonesian has been conducted by [4]. In this study, they are using the Bag of Word model feature extraction method, namely n-gram word and n-gram character. As well as comparing the performance of 4 machine learning algorithms namely, Bayesian, Logistic Regression, Naive Bayes, Support Vector Machine and Random Forest Decision Tree. But Bag of Word models also have problems in extracting the semantic meaning of a sentence. Various feature extraction methods have been proposed, including single words, single-character N-gram, multi-word N-gram, and lexical syntactic. However, semantic features between words are rarely considered in text classifications. Grammatical features can reveal deep and implicit semantic relationships between words which can be more useful in text classification [7].

The LSTM method has been used in research [8] and [9] to analyze sentiments on tweets that have better results compared to conventional methods. This shows that the LSTM method is suiTable for text classification.

Based on the background description that has been explained, this study conducted an in-depth learning approach to analyze hate speech in tweets using the Bidirectional Long Short Term Memory method and the Word Embedding approach namely Word2vec as feature extraction. With the ability of the LSTM is outstanding in doing classification and prediction on the form of time series data with unknown duration of time [10].

2. METHODS

2.1 General Architecture System

The system architecture that will be built has four supporting parts: data collection, preprocessing, feature extraction and finally classification and evaluation. The architecture to be designed in this study can be seen in Figure 1.

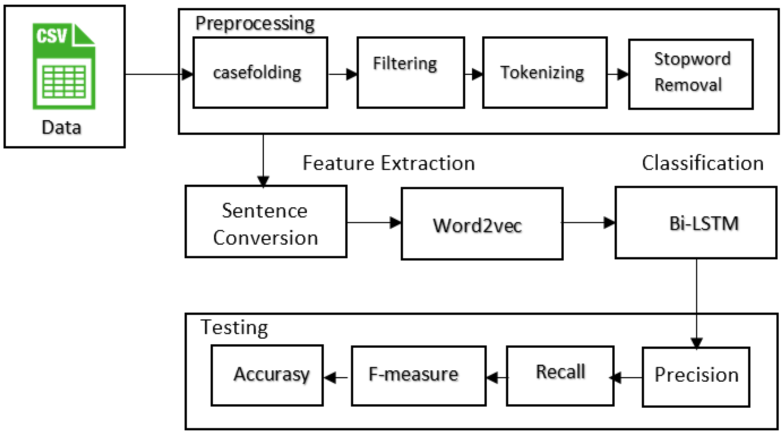

Figure 1. General Architecture System

2. 2. Data Collection

The tweet data used in this study uses a dataset from research [4]. The dataset used is Indonesian-language tweet data that have been labelled as hate speech (HS) and non-hate speech (Non_HS).

2. 3. Preprocessing

Preprocessing is to manage tweet data to get data that is cleaner than noise to facilitate the next process. The preprocessing stage consists of several steps, including 1) Case folding, 2) Filtering, 3) Tekonizing, and 4) Stopword Removal. At this stage a uniform word is made in a tweet into a lower case, filtering is used to delete special characters that are often found in tweets such as hashtag (#), @user, retweet (RT). It also removes punctuation (‘,’ ‘.’ ‘?’ ‘!’, Etc.), numeric digits (0 ... 9), and other characters (‘$’,’%’, ‘*’, etc.). Tekonizing does the process of separating words in one sentence into tokens, where spaces separate each word in one sentence. Stopword Removal functions to eliminate words that have no influence (which is, and, or, to, from, etc.). Examples of the preprocessing stage can be seen in Table 1:

Table 1 Preprocessing Example

Preprocessing	**Text**
@saidaqil apa maksud pernyataan. Anda? Apa anda mengaminkan kriminalisasi Ahok?? Hati2 jg dg mulutmu pak!	
Case Folding	@saidaqil apa maksud pernyataan. anda? apa anda mengaminkan kriminalisasi ahok?? hati2 jg dg mulutmu pak!
Filtering	saidaqil maksud pernyataan anda anda mengaminkan kriminalisasi ahok hati2 mulutmu pak
Tokenizing	[saidaqil] [apa] [maksud] [pernyataan] [anda] [apa] [mengaminkan] [kriminalisasi] [ahok] [hati2] [jg] [dg] [mulutmu] [pak]
Stopword Removal	[saidaqil] [maksud] [pernyataan] [anda] [mengaminkan] [kriminalisasi] [ahok] [hati] [mulutmu] [pak]

2. 4. Sentence Conversion

Stages carried out in sentence controversy, namely making word dictionaries, converting sentences into numbers, and padding. The results of this sentence conversion process will be used as input to the BiLSTM method. The first process is to create a word dictionary that is used to provide the word id contained in a sentence in the tweet data that has gone through preprocessing.

The second process is creating a word dictionary. Separate sentences into units of words, delete duplicate words and give values to words contained in the corpus.

2. 5. Word2vec

The word2vec architecture used in this study is the CBOW (Continuous bag-of-words) architecture and skip-gram. With a vector size of 200 dimensions with windows size 5 for CBOW architecture and 5 for skip-gram architecture. Larger windows tend to capture more information about the topic of the sentence, while smaller windows tend to attract the relationship about the word itself as an equal word or word synonym [11].

2. 6. Bidirectional LSTM

Bidirectional Long Short Term Memory is a neural network of Long Short Term Memory (LSTM) which consists of two layers of LSTM neural networks, namely the advanced LSTM layer to model the previous context and the backward LSTM layer to model each subsequent context [12]. Bidirectional LSTM is by connecting two hidden layers from opposite directions to the same output. With this generative form of deep learning, layers of neurons can obtain information from past and future conditions simultaneously. The classification and validation process is shown in Figure 2.

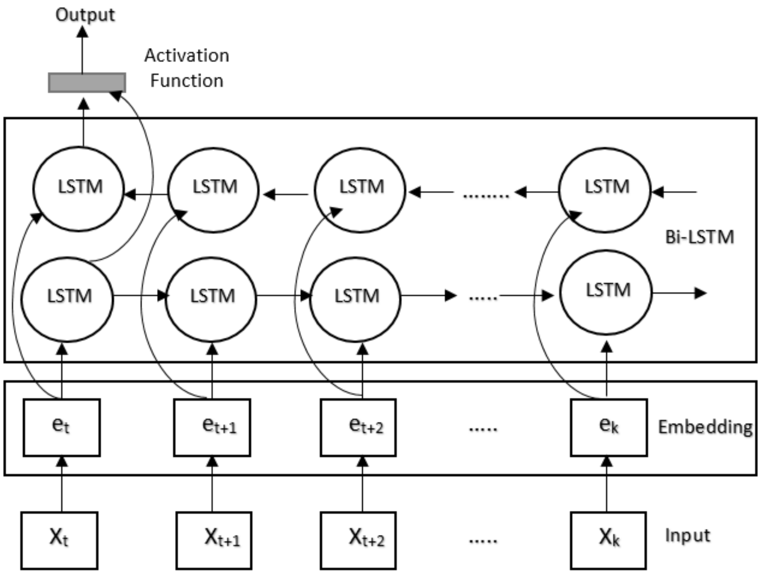

Figure 2. LSTM Classification Architecture that was built

2. 6. 1. Input Layer

The first layer is the Indonesian tweet input layer which has been changed in vector form. In the LSTM layer, several LSTM units will be tested. The LSTM unit is a memory cell that consists of four main components: input gate, self-recurrent connection, forget gate and output gate. In this study, two layers are used, namely forward and backwards, so that the output layer will get the information of the past and the future simultaneously.

2. 6. 2. Embedding

The next layer is the embedding layer (e_t). The purpose of this layer is to study the mapping of each word in the word dictionary into a vector with a lower dimension. This layer will change the positive integer index in the input into a fixed-size vector based on the vector dimensions of the word dictionary based on the word2vec model.

2. 6. 3. LSTM Layer

In LSTM, this layer determines the previous input, whether it can pass in the cell state or not. What determines the data can be continued or not is the sigmoid layer called "forget gate" (ft). Output 1 means "let it pass" and 0 means "forget information". The forget gate value can be calculated by Equation (1)

$$ft = \sigma(Wf \,. [h_{t-1}, x_t] + b_i) \tag{1}$$

The next step is the sigmoid (σ) layer called the input gate which determines which part will be updated, the tanh layer which creates a new candidate value vector ($\tilde{ct}$), which can be added to the cell state ($\tilde{ct}$). To calculate the input gate values with Equation (2) and new candidates with equations (3).

$$i_t = (W_i \, [h_{t-1}, X_t] + b_i) \tag{2}$$

$$\mathrm{C}_t = tanh(W_i \, [h_{t-1}, X_t] + b_i) \tag{3}$$

After we get the gate input values and new candidates, it is time for us to update the old C_{t-1} context to the new C_t context. with equation (4) as follows

(4)

$$Ct= ft* Ct-1+ it* \tilde{C}t$$

The final step we will get what we are looking for is results. First, we run the sigmoid gate, called the output gate, to decide what parts of the context we will produce. The context goes through tanh to make the value between −1 and 1, and we multiply it by the sigmoid gate output. The gate output can be calculated by equations (5) and (6).

$$O_t = \sigma (W_o . [h_{t-1}, x_t] + b_o \quad (5)$$

$$h_t = O_t * \tanh(C_t) \quad (6)$$

3. RESULTS AND DISCUSSION

This testing method will discuss the testing phase of the system being built. Testing is done by measuring system performance which includes testing the accuracy, precision, recall, and f-measure of the results of the classification of tweets conducted by the system. The tweet data used for this test was taken from previous research data, namely, research from [1] with a total of 713 tweets. The data is divided into two classes, namely, hate speech (HS) or 260 hate tweets (HS) and non-hate speech or non-hate speech (N_HS) of 453 tweets. Testing uses k-fold cross-validation with a k value of 10.

This testing phase is carried out to get the best accuracy results among the parameters tested. Not all combination of settings that can be done due to limited time and resources, the parameters tested and their values can be seen in Table 2.

Table 2 The parameters tested

Neuron	Epoch	L2 Regularization
10	10	0,1
20	20	0,01
30	30	0,001
40	40	
50	50	
60	60	
70	70	
80	80	
90	90	
100	100	
120	120	
200	200	

3.1 Word2vec Architecture Testing

The first test is to test the word2vec architecture used, namely Continuous bag-of-word (CBOW) and skip-gram types. Meanwhile, to compare with other methods besides the word2vec process, one hot encoding method will be tested. This test aims to determine the best kind of word2vec and one hot encoding models.

Table 3 Testing Architecture Word2vec

Arsitektur	Time (minutes)	Accurasy (%)	Precision (%)	Recall (%)	F-measurement (%)
CBOW	**09:49**	**93,00**	**98,28**	**94,04**	**96,07**
Skipgram	06:19	90,01	97,76	89,94	93,57
One Hot Encoding	03:21	74,13	79,16	73,98	76,43

Based on the results shown in Table 3, it can be seen that the CBOW type has an accuracy of 93%, better than the Skipgram type with an accuracy difference of 2.99%. While One Hot Encoding only obtained an accuracy of 74, 13%. This is because One Hot Encoding cannot represent the semantic meaning of existing words, One Hot Encoding only counts the number of vectors. When compared between CBOW and One Hot Encoding the difference in accuracy is 18.87%. CBOW Word2vec can produce better word embedding because CBOW can pay attention to the semantic meaning of each word.

3.2. Testing the Number of Neurons Against the Bi_LSTM Method

Table 4 The results of testing the number of neurons against Bi-LSTM

Number of neurons	Time (minutes)	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)
10	01:43	83,62	94,94	93,83	94,17
20	01:44	86,33	96,91	86,16	91,07
30	01:56	88,41	97,37	88,75	92,76
40	03:07	88,76	97,49	88,94	92,93
50	02:44	89,54	97,65	89,66	93,40
60	04:30	90,35	97,94	90,10	93,78
70	04:40	90,98	97,85	92,07	94,81
80	05:00	91,21	97,94	92,20	94,92
90	05:25	92,68	98,02	93,55	95,68
100	05,65	93,00	98,28	94,04	96,07
120	05:80	93,89	98,69	93,93	96,22
150	06:20	94,39	98,94	93,62	96,18
200	**07:35**	**94,66**	**99,08**	**93,74**	**96,29**
220	08:20	70,67	92,53	66,94	76,71
250	09:30	71.34	92,94	65,99	76,34
270	10:15	70,54	92,74	66,02	76,21
300	12:45	70,19	92,17	67,02	76,41

In this test was given the number of Epoch 10 and L2 value of 0.001. According to table 4 It can be seen that the number of neurons up to 50 accuracy value obtained can not provide significant improvement in accuracy, the number of neurons continued to add up to 300 neurons, when the number of neurons 200 earned accuracy value of 94.66%, when the number of neurons 220 to 300 there is a decrease in the accuracy value of 24.47%, the number of neurons 300 gets an accuracy value of 70.19%. From the results of this experiment obtained the highest accuracy value when the number of neurons 200 is 94.66%. The more the number of neurons, the longer the computing time is needed. On the other hand, a large number of neurons does not guarantee that it can increase the significant accuracy, precision, recall, and F-measurement values.

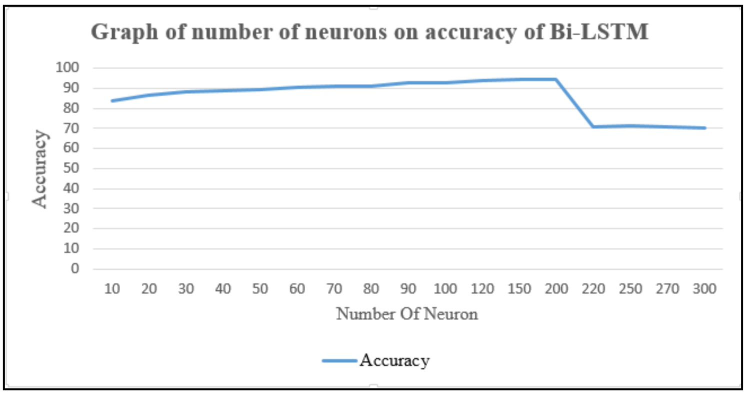

Figure 3. Graphic Effect of the Number of Neurons on Bi-LSTM Accuracy

During the test, there is no specific formula that can determine the optimal number of neurons for increasing accuracy in the model created. Comparison of the accuracy of the number of neurons to the Bi-LSTM method can be seen in Table 5.

Table 5 Comparison of accuracy of the number of neurons

Number of Neuron	LSTM (%)	BI-LSTM (%)	BI-LSTM 3 Layer (%)
10	74,49	83,62	87,38
20	84,03	86,33	89,65
30	84,28	88,41	90,91
40	85,94	88,76	92,58
50	86,78	89,54	94,36
60	87,00	90,35	95,03
70	87,25	90,98	95,93
80	87,24	91,21	96,01
90	87,64	92,68	95,75
100	87,64	93,00	96,15
120	88,22	93,89	96,32
150	88,45	94,39	**96,93**
200	**88,78**	**94,66**	94,55

From the test results above we can see the accuracy of the LSTM method to the number of neurons, the best results are found in the Bi-LSTM practice with 3 layers with an accuracy of 96.93%. The best amount of neurons can be used for further parameter testing.

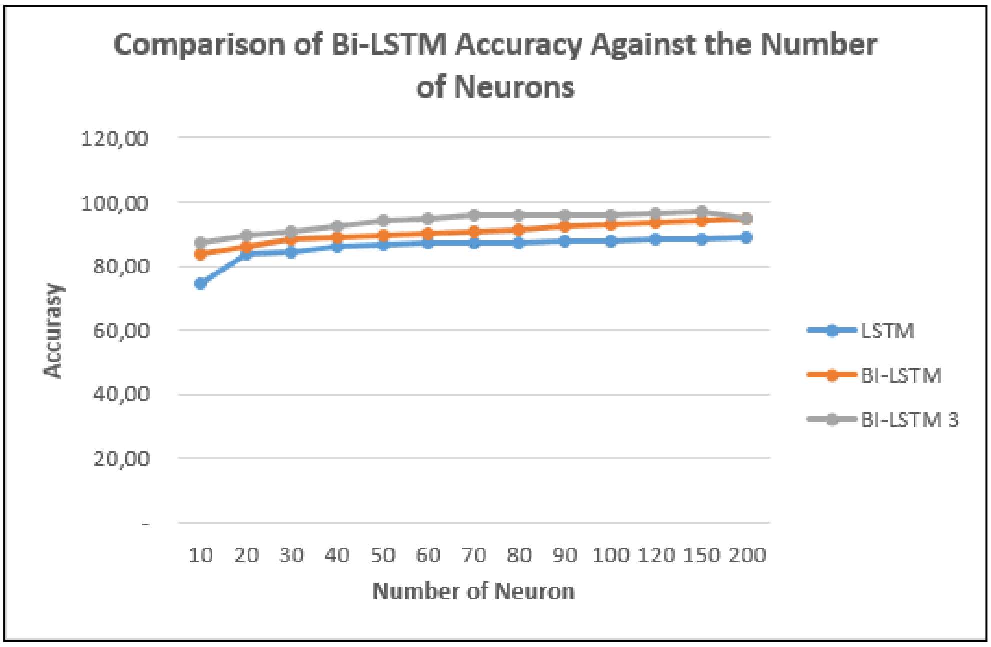

Figure 4. Comparison of accuracy of the number of neurons

After testing the number of neurons to get the best amount of neurons to the Bidirectional Long Short Term Memory (Bi-LSTM) method with epoch 10 parameters and the number of neurons 10 to 200, it can be seen that the Bi-LSTM process with 3 layers gets the best accuracy namely 96.93% with a total of 200.

3.3. Testing the Amount of Epoch Against Bi_LSTM Method

In previous tests, the number of neurons has been determined based on test results that show the best value. In this study, the number of epoch tested was 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 while the number of neurons was 200 because it had the highest accuracy based on previous testing and an L2 value of 0.001. In Table 5 it can be seen that the number of epoch 100 gives the results with the best accuracy.

Table 6 The results of testing the number of epoch

Number of Epoch	Time (minutes)	Accurasy (%)	Loss
5	04:02	63,89	0,65
10	08:47	64,38	0,61
20	10:05	86,72	0,43
30	13:09	87,63	0,3
40	15:23	86,83	0,25
50	17:15	92,97	0,2
60	17:55	90,24	0,16
70	18:20	91,60	0,22
80	18:57	96,63	0,11
90	19:10	97,87	0,08
100	19:55	98,10	0,07

3.4. L2 Regularization Testing

Previous tests have obtained the word2vec architecture, the best number of neurons, and the number of epochs. Finally, determine the optimal L2 regularization value. The L2 values tested were 0.1, 0.01 and 0.001. The test results can be seen in Table 7.

Table 7. L2 Regularization test results

L2	Time (minutes)	Accurasy (%)	Precision (%)	Recall (%)	F-measurement (%)
0.1	08:37	90,87	97,97	91,14	94,31
0.01	04:30	90,90	98,01	91,26	94,39
0.001	03:18	91,10	98,18	90,69	94,19

The value of L2 is directly proportional to the length of time required. However, the magnitude of the L2 value can increase accuracy but not significantly.

3.5. Bi-LSTM Performance Testing

Based on previous tests obtained Word2vec architecture, namely CBOW, the number of neurons, the number of epochs, and the best L2 Regularization value. The best results from previous tests will be used to test the performance of the Bi-LSTM method and will be compared with the LSTM method, the Bi-LSTM 3 layer. The test results are shown in Table 8 of the BiLSTM test results.

Table 8 Comparison Results of BiLSTM Testing

	Accurasy (%)	**Precision (%)**	**Recall (%)**	**F-measure (%)**
Bi-LSTM	94,66	99,08	93,74	96,29
Bi-LSTM 3 Layer	**96,93**	**99,74**	**94,61**	**97,02**
LSTM	88,78	96,92	91,74	94,16

From the test results above we can see that the Bi-LSTM 3 layer method gets the highest accuracy value which is 96.93%. This shows that the addition of hidden layer to the LSTM layer can increase the accuracy value in the LSTM method even though the increase in accuracy is not too significant, but adding this layer will take a long time to process the test.

4. CONCLUSIONS

The use of Word2vec and the Bidirectional method of Long Short Term Memory with CBOW architecture, with epoch 10, learning rate 0.001 and the number of neurons 200 on the hidden layer, generates an accuracy rate of 94,66%, with each precision value of 99,08%, 93,74% recall and F-measure 96,29%. As for the Bidirectional Long Short Term, Memory with three layers has 96,93% accuracy. The addition of one layer to BiLSTM increased by 2,27%.

However, in this study, there are still shortcomings that can be added to the training data aside from the Wikipedia data training in Indonesian to avoid words that cannot be represented.

REFERENCES

[1] A. K. B. A. Putra, M. A. Fauzi, B. D. Setiawan, and E. Setiawati, “Identifikasi Ujaran Kebencian Pada Facebook Dengan Metode Ensemble Feature Dan Support Vector Machine,” *J. Pengemb. Teknol. Inf. Dan Ilmu Komput.*, vol. 2, no. 12, 2018.

[2] M. M. Munir, M. A. Fauzi, and R. S. Perdana, “Implementasi Metode Backpropagation Neural Network berbasis Lexicon Based Features dan Bag of Words Untuk Identifikasi Ujaran Kebencian Pada Twitter,” *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 2, no. 10, pp. 3182–3191, 2018.

[3] G. A. Buntoro, “Analisis Sentimen Hatespeech Pada Twitter Dengan Metode Naïve Bayes Classifier Dan Support Vector Machine,” *J. Din. Inform.*, vol. 5, no 2, September 2016.

[4] I. Alfina, R. Mulia, M. I. Fanany, and Y. Ekanata, “Hate Speech Detection in the Indonesian Language : A Dataset and Preliminary Study,” in *9th Int. Conf. Adv. Comput. Sci. Inf. Syst. (ICACSIS 2017)*.

[5] N. Chetty and S. Alathur, “Aggression and Violent Behavior Hate speech review in the context of online social networks,” *Aggress. Violent Behav.*, vol. 40, no. March 2017, pp. 108–118, 2018.

[6] M. C. Anam and M. Hafiz, “Surat Edaran Kapolri Tentang Penanganan Ujaran Kebencian (Hate Speech) dalam Kerangka Hak Asasi Manusia,” *J. Keamanan Nas.*, vol. I, 2015.

[7] Z. Su, H. Xu, D. Zhang, and Y. Xu, “Chinese sentiment classification using a neural

network tool—Word2vec," in *Multisensor Fusion and Information Integration for Intelligent Systems (MFI), 2014 International Conference on*, pp. 1–6, 2014.

[8] D. Li and J. Qian, "Text Sentiment Analysis Based on Long Short-Term Memory," *2016 First IEEE Int. Conf. Comput. Commun. Internet*, pp. 471–475, 2016.

[9] A. Hassan and A. Mahmood, "Efficient Deep Learning Model for Text Classification Based on Recurrent and Convolutional Layers Efficient Deep Learning Model for Text Classification Based on Recurrent and Convolutional Layers," *IEEE Access*, no. February, 2018.

[10] E. Kang, "Long Short-Term Memory (LSTM): Concept," 2017. [Online]. Available: https://medium.com/@kangeugine/long-short-term-memory-lstm-concept-cb3283934359. [Accessed: 18-Jan-2019].

[11] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in neural information processing systems*, pp. 2177–2185, 2014.

[12] A. M. Ertugrul and P. Karagoz, "Movie Genre Classification from Plot Summaries Using Bidirectional LSTM," *Proc. - 12th IEEE Int. Conf. Semant. Comput. ICSC 2018*, vol. 2018-Janua, pp. 248–251, 2018.