Emotion Detection in Twitter Social Media Using Long Short-Term Memory (LSTM) and Fast Text

M. Alfa Riza[1]*, Novrido Charibaldi[2]
[1,2]Informatics Department, Universitas Pembangunan Nasional Veteran Yogyakarta, Indonesia
[1]malfariza45@gmail.com; [2]novrido@upnyk.ac.id
*Corresponding author

ABSTRACT

Emotion detection is important in various fields such as education, business, employee recruitment. In this study, emotions will be detected with text that comes from Twitter because social media makes users tend to express emotions through text posts. One of the social media that has the highest user growth rate in Indonesia is Twitter. This study will use the LSTM method because this method is proven to be better than previous studies. Word embedding fast text will also be used in this study to improve Word2Vec and GloVe that cannot handle the problem of out of vocabulary (OOV). This research produces the best accuracy for each word embedding as follows, Word2Vec produces an accuracy of 73,15%, GloVe produces an accuracy of 60,10%, fast text produces an accuracy of 73,15%. The conclusion in this study is the best accuracy was obtained by Word2Vec and fast text. The fast text has the advantage of handling the problem of out of vocabulary (OOV), but in this study, it cannot improve the accuracy of word 2vec. This study has not been able to produce very good accuracy. This is because of the data used. In future works, to get even better results, it is expected to apply other deep learning methods, such as CNN, BiLSTM, etc. It is hoped that more data will be used in future studies.

Keywords: Deep Learning; Emotion Detection; LSTM; Fast Text

Article History

Received : May, 8th 2021	Revised : May, 31th 2021	Accepted : May, 31th 2021

I. INTRODUCTION

Emotion detection has many uses in making decisions, such as in the field of education to predict the emotional state of students with an e-learning platform so that it can be used to improve the quality of learning [6]. The emotion detection business field can be used to review product quality [19]. Emotion detection is also useful for employee recruitment, counseling, and psychological profiling because it is part of personality predictions [5].

Emotions can be detected through voice, facial expressions, hand movements, body movements, heart rate, blood pressure, text [4]. Text is relatively easier to use for emotion detection because emotions are triggered by certain situations and describe emotions from situations without words [2]. A very popular emotion model is the Ekman emotion model. Ekman's model divides emotions into six emotional labels, namely happiness, anger, fear, disgust, sadness, and surprise [31]; these labels are universal in different cultures.

Currently, social media makes users tend to express emotions through text posts. One of the social media that has the highest user growth rate in Indonesia is Twitter; active Twitter users in Indonesia occupy the third position in the Asia Pacific from 2012 to 2018 [28]. Research result [8] using the Naïve Bayes method with a combination of N-gram features, but this study only produced the highest accuracy of 55.54%. The research [24] resulted in a less than maximum accuracy of 55.54% from 34,872 words for lexicon *with* EmoLex, while NLP produced only 61.53% accuracy. The Term Frequency-Inverse Document Frequency (TF-IDF) only produced 59% accuracy of 6 emotions, but if only one emotion produces a fairly good accuracy of 87.23% [23]. The Support Vector Machine (SVM) and K-Nearest Neighbor methods [1], where SVM produces 45.64% precision, 50.20% recall, and pretty good accuracy of 81.04%, while for KNN it produces less maximum precision, recall, and accuracy, that's 34.21% precision, 45.95% recall. 45,95%, an accuracy of 79.70%. The research [13] compares the SVM, Long Short-Term Memory (LSTM), and Nested Long Short-Term Memory (NLSTM) methods. SVM produces accuracy, precision, lowest recall, namely accuracy 98.679%, precision 98.53%, recall 98.22%, while NLSTM obtains the highest accuracy with word embedding Word2Vec, resulting in 99.167% accuracy, 99.21% precision, 98.83 recall. LSTM obtained %, but the highest precision and recall with word embedding Word2Vec resulting in 99.22% precision, 98.86% recall, while the accuracy was not much different from NLSTM, which was 99.154%. The research [7] using Long Short-Term Memory (LSTM) with word embedding Glo-Ve produces an accuracy of only 50%, precision is only 33%, recall is only 38% because, in the training process, underfitting

DOI: 10.25139/ijair.v3i1.3827

occurs. Previous research has shown that NLSTM has the best accuracy, but LSTM has better precision and recall than NLSTM, which means LSTM has better performance. LSTM is effective for processing sequential data such as text because it has memory cells [17].

LSTM with Word2Vec word embedding is widely used because Word2Vec can recognize similar words. Glo-Ve word embedding is also widely used because it can obtain semantic relationships between words and co-occurrence [30], but Word2Vec and Glo-Ve cannot handle typographical errors, thus reducing accuracy [16]. This happens because Word2Vec and Glo-Ve cannot represent vectors of words that have never been encountered; this problem is called out of vocabulary (OOV). Word embedding fast text can handle OOV problems by paying attention to sub word information using n-gram [3]. So, this research uses the Long Short-Term Memory (LSTM) method and uses word embedding fast text.

This study will apply Long Short-Term Memory (LSTM) with word embedding fast text to detect emotions with six emotional labels: happiness, anger, fear, disgust, sadness, and surprise. This study will use data sourced from Twitter. This research is expected to produce an application that can classify emotions into happiness, anger, fear, disgust, sadness, and surprise, which has better accuracy. The results of emotional detection can be helpful to in making decisions in various fields such as education, business, recruitment. Employees.

II. METHOD

The flow of the research method in this study can be seen in Fig. 1, and the research method contains the steps taken in the study, from data collection to testing, so that finally, the research conclusions can be drawn.

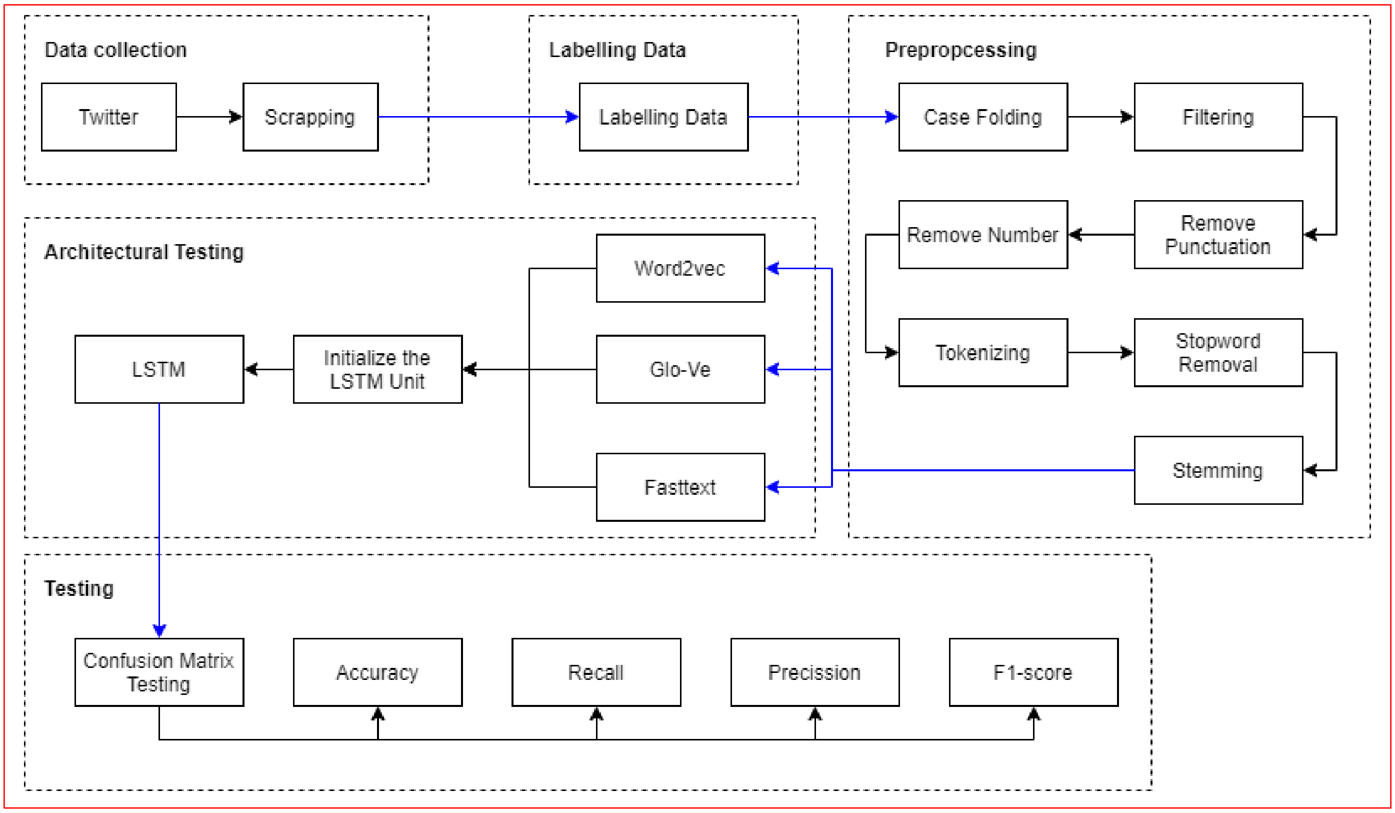


Fig. 1. The Research Method

A. Data Collection and Labelling Data

The dataset used in this study came from Twitter, which was taken from several influencers as the primary data and trending data for one week as supporting data. Influencers were chosen because becoming influencers required a self-disclosure process in social media, where one form of self-disclosure is to express emotions [26]. The data was taken by using the web scraping technique using python programming language with selenium library and producing a dataset with **.csv/*.xlsx* file format. The number of data is 1304 with 250 for happy data, 250 for sad data, 200 for scared data, 200 for disgust data, 204 for angry data, 201 for shocking data. Next, data were labeled by students who actively use Twitter. This was done so that the data obtained for the training model was not subjective. An example of the dataset in this study can be seen in Table I.

TABLE I
EXAMPLE OF THE DATASET

Text	Emotion
emang cocok banget mukanya senyum terus bahagia selalu	happiness
Sedih banget pas pamitan ke bapak kos	sadness
Kalau ketemu anjing kabur aja, takut	fear
dih jijik banget lu	disgust
Dah lah bikin emosi	anger
Terkejut bila tengok Berita tadi	surprise

B. *Preprocessing*

The first step after labeling data is preprocessing. Data preprocessing often affects the performance of machine learning, making it more effective [25]. Preprocessing is useful for cleaning data from noise and uninformative parts that are not needed so that the text is ready to be classified [10]. Preprocessing will make the data clean so that the process of creating word vectors and classifications is more accurate [20]. In this study, preprocessing used was case folding, remove punction, remove a number, tokenizing, stop removal, stemming:

- Case Folding is changing all letters in the text to lower case or lowercase all letters [27], with letters received from "a" to "z". Case folding aims to provide standardization in writing.
- Remove punctuation is to remove punctuation in the text to reduce the burden on classification processing because it is considered unimportant and includes delimiters, examples of punctuation that are removed by period (.), Comma (,), question mark (?), Slash (/), hashtag (#), exclamation point (!) and others.
- Remove numbers is to remove numbers in a text; this deletion is because numbers are considered meaningless and include delimiters, similar to remove punctuation, only different in objects that are deleted.
- Tokenizing or tokenization is cutting sentences based on each constituent word, usually cutting based on whitespace such as spaces, tabs, and enter. Each tokenized word is called a token [15].
- Stop word removal or filtering is a process to remove words that have no meaning or meaning but will not change the meaning of these comments [15]. Stop word removal will reduce index size, classification processing time, and noise from data. Usually, stop words are in the form of pronouns and conjunctions, such as "aku", "kamu", "kita", "dan", "atau". In this study, the process of case folding, remove punctuation, remove numbers, and stop words use the Natural Language Toolkit (NLTK) library, with an Indonesian stop word.
- Stemming or lemmatization is a process for transforming verbs that adhere to a document into a root word by removing prefixes, suffixes, and insertions—stemming aims to reduce word variants with almost the same meaning in a document improve performance at the information retrieval stage.

In this study stemming process will use library sastrawi because the data in this study use Indonesian. The library sastrawi applies the Nazief and Adriani algorithms. Examples of preprocessed data can be seen in Table II.

TABLE II
EXAMPLE OF THE DATASET

Preprocessing Process	Text
Text (before *preprocessing)*	*"Hore, akhirnya timnas Indonesia menang dengan skor 2-0 !!"*
Text after case folding	*"hore, akhirnya timnas indonesia menang dengan skor 2-0 !!"*
Text after removing punctuation	*"hore, akhirnya timnas indonesia menang dengan skor 20"*
Text after remove number	*"hore akhirnya timnas indonesia menang dengan skor"*
Text after *tokenizing*	*["hore", "akhirnya", "timnas", "indonesia", "menang", "dengan", "skor"]*
Text after stop word removal	*["hore", "akhirnya", "timnas", "indonesia", "menang", "skor"]*
Text after *stemming*	*["hore", "akhir", "timnas", "indonesia", "menang", "skor"]*
Text after preprocessing	*["hore", "akhir", "timnas", "indonesia", "menang", "skor"]*

C. Word Embedding

The results of the preprocessing then enter the word embedding process. Word embedding is a technique for mapping words based on an existing dictionary to produce numeric vectors containing real numbers [11]. Research conducted by Utomo (2020) found that the LSTM algorithm has better accuracy when using word embedding of 86.76%, while without word embedding only gets an accuracy of 84.14% [29]. In this study, we will test 3 embedding words, namely, Word2Vec, GloVe, FastText. Word embedding testing is useful so that at the end of the study, conclusions can be drawn, whether FastText can improve accuracy or not. In this study, Word2Vec and FastText are implemented using the Gensim library, while GloVe will be implemented using pre-trained GloVe. Each word embedding uses 100 dimensions.

Word2Vec is one of the word embedding methods introduced by Mikolov et al. In 2013. Word2Vec has advantages in the similarity of word meanings obtained from paying attention to the similarity of words around the target word [16]; the Word2Vec method is very popular because of the advantages. Word2Vec has two techniques, namely the continuous bag of words and the skip-gram model. In this study, the Word2Vec model used is CBOW. Examples of Word2Vec results with a 5-dimensional vector with the same sentence as the results of the preprocessing process examples can be seen in Table III.

TABLE III
EXAMPLE OF WORD2VEC

Words	Dimensions				
	D1	D2	D3	D4	D5
hore	0.084777	0.057706	-0.082217	-0.054878	-0.096413
akhir	-0.094585	0.068689	0.090819	-0.059799	0.088703
timnas	-0.076862	-0.039967	0.068271	0.027309	-0.035721
indonesia	0.095575	-0.025707	-0.084364	-0.094829	-0.006755
menang	0.085284	-0.021515	-0.033860	-0.067008	-0.096384
skor	-0.022820	0.086830	-0.074412	-0.014862	0.085389

Global vectors for word representation (GloVe) are a word embedding method that relies on the co-occurrence of words or statistics on the occurrence of words in a set of words or corpus, which are captured directly by the model to obtain semantic relationships between words in the corpus. Glo-Ve uses the global matrix factorization method, representing the number of occurrences or frequencies in a corpus [21].

Fast text is a word embedding method developed from Word2Vec. Fast Text has the advantage that it can handle the out of vocabulary problem where this problem cannot be solved by Word2Vec and Glo-Ve word embedding. Fast text learns word representation by paying attention to sub-word information using n-grams into the skip-gram model. This makes fast text capture shorter words and understands word suffixes and prefixes [3]. Examples of FastText results with a 5-dimensional vector with the same sentence as the results of the preprocessing process examples can be seen in Table IV.

TABLE IV
EXAMPLE OF FASTTEXT

Word	Dimensions				
	D1	D2	D3	D4	D5
hore	-0.015188	0.006831	-0.000868	-0.033525	-0.036368
akhir	0.026232	-0.020445	0.043318	0.081208	-0.070539
timnas	0.041364	0.038119	-0.043051	-0.017365	-0.014987
indonesia	0.026756	-0.005694	-0.003622	-0.020880	0.005214
menang	-0.005905	0.024995	0.015284	0.050226	0.000018
skor	0.041438	-0.059298	0.087633	0.081472	0.033918

D. Long Short-Term Memory (LSTM)

The results of the word embedding then enter into the process of models the LSTM model. LSTM is a deep learning algorithm developed from the RNN architecture. RNN has a vanishing gradient problem; LSTM can solve the problem of vanishing gradient with memory cells and gate units (input gate, forget gate, output gate) so that LSTM can read, store, and update information [22]. In this study, LSTM was implemented using library Keras. LSTM architecture can be seen in Fig. 2.

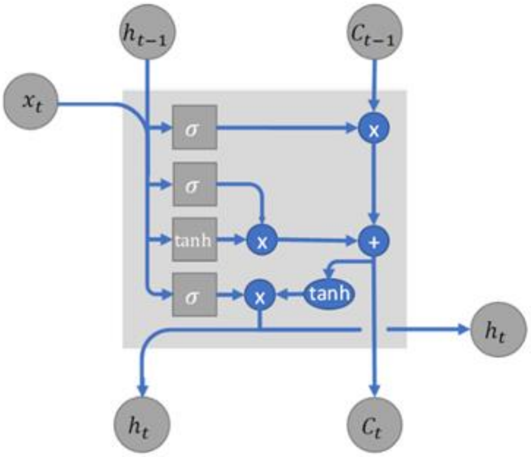


Fig. 2. LSTM Architecture [17]

There are three types of gates on the LSTM: forget gate, input gate, and output gate. The forget gate serves to determine which information is removed from the cell. The input gate functions to determine the value of the input to be updated on state memory. The output gate functions to determine the output based on the input and memory in the cell [29]. LSTM steps are as follows [14]:

- The first step, as illustrated in Fig. 3, LSTM, will determine which information should be removed from the cell state; this section is called the forget gate.

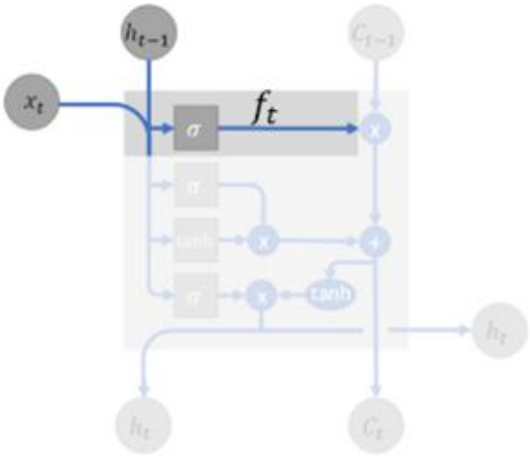


Fig. 3. Forget Gate [17]

This step uses the output from the previous step (ht-1) and input (x_t) to be processed with the sigmoid activation function. This process will produce an output in the form of a value of 0 or a value of 1 in cell state C_{t-1}, *a* value of 0 means that information will be removed, while a value of 1 means that it will be preserved. According to Hochreiter and Schmidhuber (1997), the forget gate equation is described in equation (1).

$$ft = \sigma\,(x_t\,.\,W_f\; + h_{t-1}\,.\,U_f + \;b_f) \tag{1}$$

- As illustrated in Fig. 4, the second step determines the information to be added to the cell state. This step has two parts; the first, the sigmoid layer, called the input gate layer, which functions to determine which value to update. Next, the tanh layer creates a new candidate value (C_t) to be inserted into the cell state. The output from the input gate layer and tanh layer will be combined to update the cell state. According to Hochreiter and Schmidhuber (1997), the equation for the input gate layer and the equation for the value of the candidate context can be seen in equation (2) and equation (3).

$$i_t = \sigma\,(x_t\,.W_i + \;h_{t-1}\,U_i + \;b_i) \tag{2}$$

$$\check{C}_t = \tanh\,(x_t\,.W_c + \;h_{t-1}.U_c + \;b_c) \tag{3}$$

E -ISSN : 2686-6269

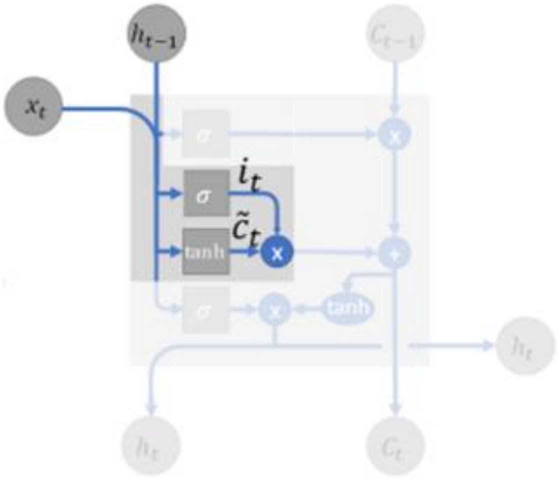


Fig. 4. Input Gate And Creates A New Candidate Value [17]

- As illustrated in Fig. 5, the third step is updating the old state cell C_{t-1} to become the new state cell C_t. In the previous step, the input gate value and the candidate context were obtained.

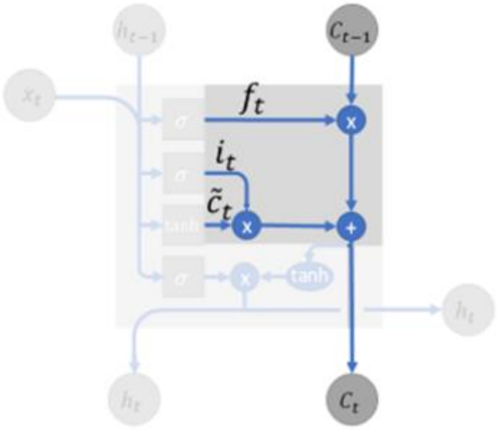


Fig. 5. Updating The Old State Cell [17]

The third step will occur the process of multiplying the old C_{t-1} cell state with the ft forget gate value, then the multiplication result of the old C_{t-1} cell state with the forget gate value plus the multiplication result of the input gate value with the context candidate value $\tilde{C}_t$, will produce a new cell state (C_t). The equation for getting a new cell state (C_t), according to Hochreiter and Schmidhuber (1997) can be seen in equation (4).

$$\mathrm{C}_t = f_t \times \mathrm{C}_{t-1} + i_t \times \check{\mathrm{C}}_t \tag{4}$$

- As illustrated in Fig. 6, the final step determines the output of the whole process (h_t). First, the sigmoid layer uses the previous output (ht-1) and input (x_t) to determine the gate output value (O_t), the gate output value (O_t) between 0 and 1, which shows the part that is the output of the cell state, then the cell state (C_t) is changed using the tanh activation function, this cell state gets a value between -1 and 1. The cell state value is then multiplied by the gate output value (O_t) to give the output value (h_t). This process equation, according to Hochreiter and Schmidhuber (1997) is described in equation (5) and equation (6).

$$O_t = \sigma\,(x_t \times W_o + h_{t-1} \times U_o + b_o) \tag{5}$$

$$h_t = O_t \times \tanh(\check{C}_t) \tag{6}$$

DOI: 10.25139/ijair.v3i1.3827

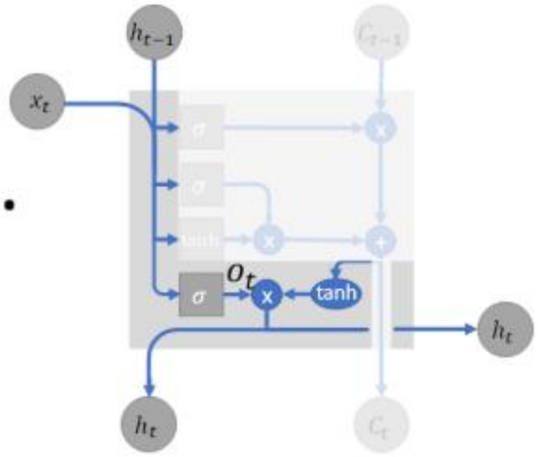


Fig. 6. Output Gate [17]

LSTM has a Dense layer; this layer is useful for determining classification results. The number of dense layers used in this study is 6, according to the number of classes used. The dense layer gets input from the final output (ht) of the last order. The Dense Layer formula can be seen in equation (7).

$$Y = \sigma(h_t \times W_y + b_y) \tag{7}$$

LSTM has several parameters, including the number of units/neurons, activation function, optimization, epoch, number of dropouts. This research will use the number of units/neurons of 50 to 200 units, while the dropout will use 20, 30, and 50. Dropout is useful for disconnecting some connections between units randomly to reduce overfitting significantly. The activation function in this study uses the sigmoid activation function. The optimization used in this study is Adaptive Moment Estimation (ADAM). The number of epochs used in this study was 50 epochs. The loss function used is the Categorical Cross-Entropy Loss function because, in this study, there are more than two classes. This study's division between training data and testing data uses 70% training data and 30% testing data. This is because the number of dividing 70:30 and 80:20 can produce the best model [9]. Testing of neuron parameters and dropout is useful for getting the best architecture.

E. Testing

The last stage is testing according to the architectural design being built and word embedding that is compared, and the test will use a confusion matrix to obtain accuracy, precision, recall, and f1-score values. References [12] confusion matrix is a tool for analyzing a classifier to recognize tuples from different classes properly or not. In this study, a confusion matrix implemented using Sklearn.Metrics. There are four terms in the confusion matrix that represent the results, namely True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN). The confusion matrix illustration table can be seen in Table V.

TABLE V
ILLUSTRATION OF CONFUSION MATRIX

		Actual Values	
		Positive	Negative
Predicted Values	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

The equations for accuracy, precision, and recall are as follows [12] :

- Accuracy is a value that represents the level of closeness between the predicted data and the actual data. The accuracy formula can be seen in equation (8).

$$accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{8}$$

where TP is several positive data that is predicted to be correct, TN is several negative data that is predicted to be correct., FP is several negative data predicted was wrong.; FN is several data positives that were predicted to be wrong

DOI: 10.25139/ijair.v3i1.3827

- Precision is a comparison between data that is predicted to be true positive with data that is predicted to be positive. The precision formula can be seen in equation (9).

$$precision = \frac{TP}{TP+FP} \quad (9)$$

- The recall is a comparison between data that is predicted to be true positive with truly positive data. The recall formula can be seen in equation (10).

$$recall = \frac{TP}{TP+FN} \quad (10)$$

III. RESULT AND DISCUSSION

Below are the results based on parameters such as the number of units/neurons of 50 to 200 units, while the dropout will use 20, 30, and 50. In this section, the word embedding is compared between Word2Vec, GloVe, and FastText. This aims to see whether FastText can affect the accuracy of results to conclude at the end of the study. The results table can be seen in Table VI, Table VII, and Tabe VIII.

TABLE VI
RESULT TEST LSTM AND WORD2VEC

Word Embedding	Unit	Dropout	Accuracy	Precision	Recall	F1-Score
Word2Vec	50	20	0.721228	0.721228	0.721228	0.721228
		30	0.705882	0.705882	0.705882	0.705882
		50	**0.731458**	**0.731458**	**0.731458**	**0.731458**
	100	20	0.685422	0.685422	0.685422	0.685422
		30	0.721228	0.721228	0.721228	0.721228
		50	0.680307	0.680307	0.680307	0.680307
	150	20	0.708440	0.708440	0.708440	0.708440
		30	0.718670	0.718670	0.718670	0.718670
		50	0.700767	0.700767	0.700767	0.700767
	200	20	0.710997	0.710997	0.710997	0.710997
		30	0.721228	0.721228	0.721228	0.721228
		50	0.705882	0.705882	0.705882	0.705882

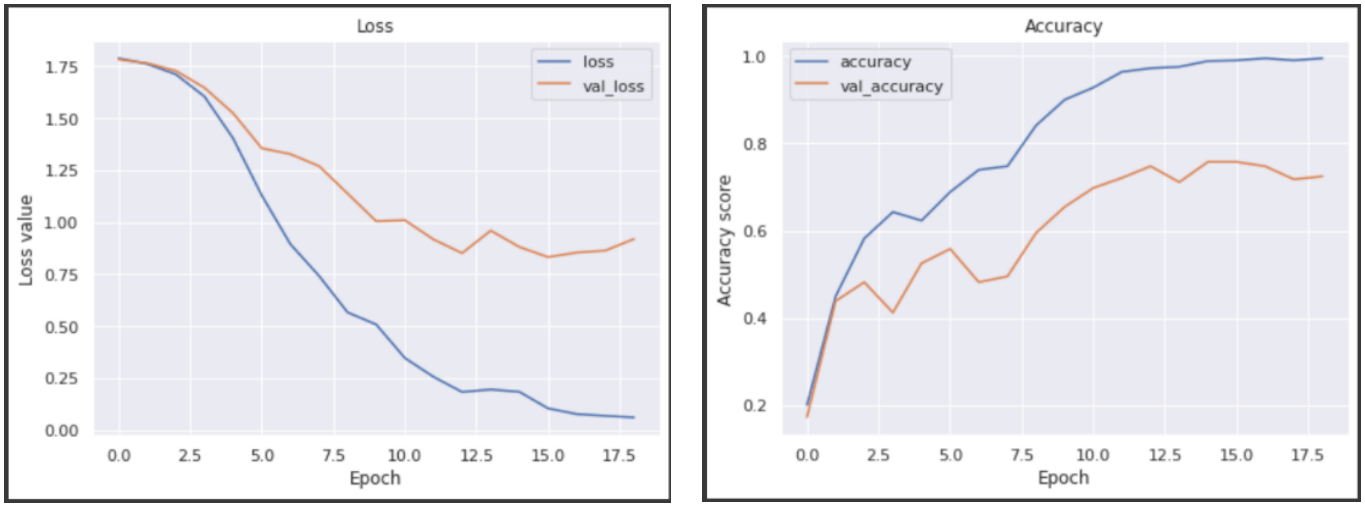
TABLE VII
RESULT TEST LSTM AND GLO-VE

Word Embedding	Unit	Dropout	Accuracy	Precision	Recall	F1-Score
GloVe	50	20	0.585678	0.585678	0.585678	0.585678
		30	**0.601023**	**0.601023**	**0.601023**	**0.601023**
		50	0.598465	0.598465	0.598465	0.598465
	100	20	0.544757	0.544757	0.544757	0.544757
		30	0.529412	0.529412	0.529412	0.529412
		50	0.557545	0.557545	0.557545	0.557545
	150	20	0.557545	0.557545	0.557545	0.557545
		30	0.521739	0.521739	0.521739	0.521739
		50	0.519182	0.519182	0.519182	0.519182
	200	20	0.529412	0.529412	0.529412	0.529412
		30	0.491049	0.491049	0.491049	0.491049
		50	0.452685	0.452685	0.452685	0.452685

TABLE VIII
RESULT TEST LSTM AND FAST TEXT

Word Embedding	Unit	Dropout	Accuracy	Precision	Recall	F1-Score
Fast text	50	20	0.721228	0.721228	0.721228	0.721228
		30	0.708440	0.708440	0.708440	0.708440
		50	**0.731458**	**0.731458**	**0.731458**	**0.731458**
	100	20	0.690537	0.690537	0.690537	0.690537
		30	0.721228	0.721228	0.721228	0.721228
		50	0.703325	0.703325	0.703325	0.703325
	150	20	0.708440	0.708440	0.708440	0.708440
		30	0.716113	0.716113	0.716113	0.716113
		50	0.695652	0.695652	0.695652	0.695652
	200	20	0.685422	0.685422	0.685422	0.685422
		30	0.710997	0.710997	0.710997	0.710997
		50	0.710997	0.710997	0.710997	0.710997

In the Word2Vec word embedding test table, the highest accuracy is obtained by the 50-unit architecture and 50 dropouts of 0,731458. In the Word2Vec training process, the relationship between loss and loss validation can be seen in Fig. 7(a), while the relationship between accuracy and accuracy validation can be seen in Fig. 7(b). Meanwhile, the results of the confusion matrix test on the Word2Vec architecture can be seen in Table IX.



(a) Loss And Loss Validation LSTM And Word2Vec

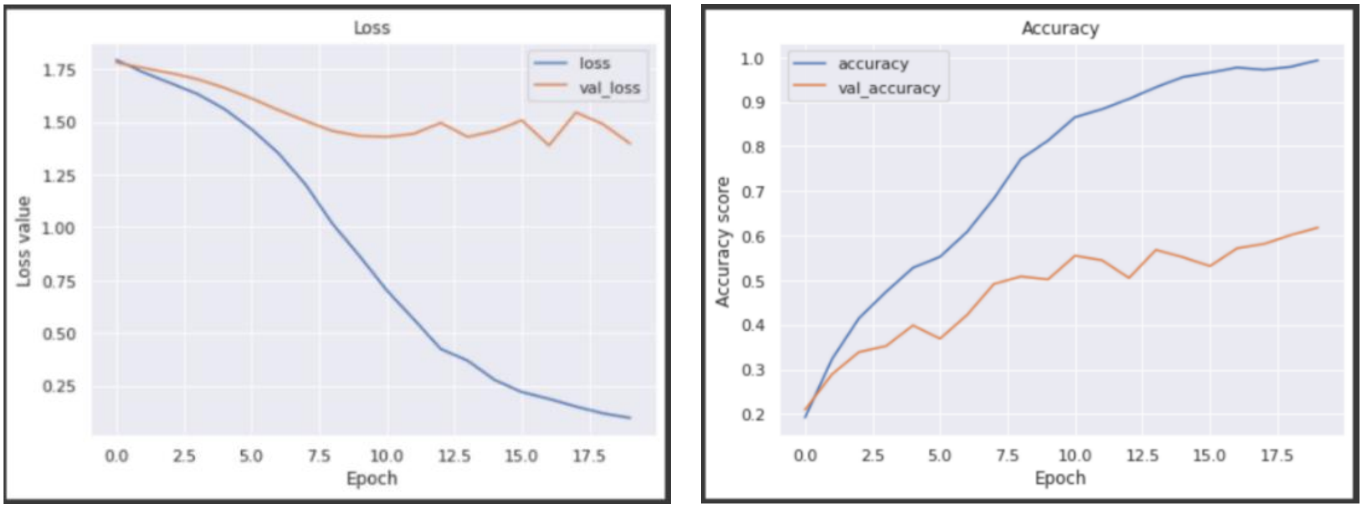
(b) Accuracy And Validation Accuracy LSTM And Word2Vec

Fig. 7. The Word2Vec Word Embedding Test

TABLE IX
RESULT CONFUSION MATRIX LSTM AND WORD2VEC

	happiness	*sadness*	*fear*	*disgust*	*anger*	*surprise*
happiness	62	4	1	0	1	6
sadness	16	55	0	1	6	4
fear	3	0	54	2	3	0
disgust	1	1	0	47	1	0
anger	4	10	1	1	38	2
surprise	12	17	0	1	7	30

In the GloVe word embedding test table, the highest accuracy is obtained by the 50-unit architecture and 30 dropouts of 0,601023. In the GloVe training process, the relationship between loss and loss validation can be seen in Fig. 8(a), while the relationship between accuracy and accuracy validation can be seen in Fig. 8(b). Meanwhile, the results of the configuration matrix test on the GloVe architecture can be seen in table X.



(a) Loss And Loss Validation LSTM And GloVe

(b) Accuracy And Validation Accuracy LSTM And GloVe

Fig. 8. The Glo-Ve Word Embedding Test Table

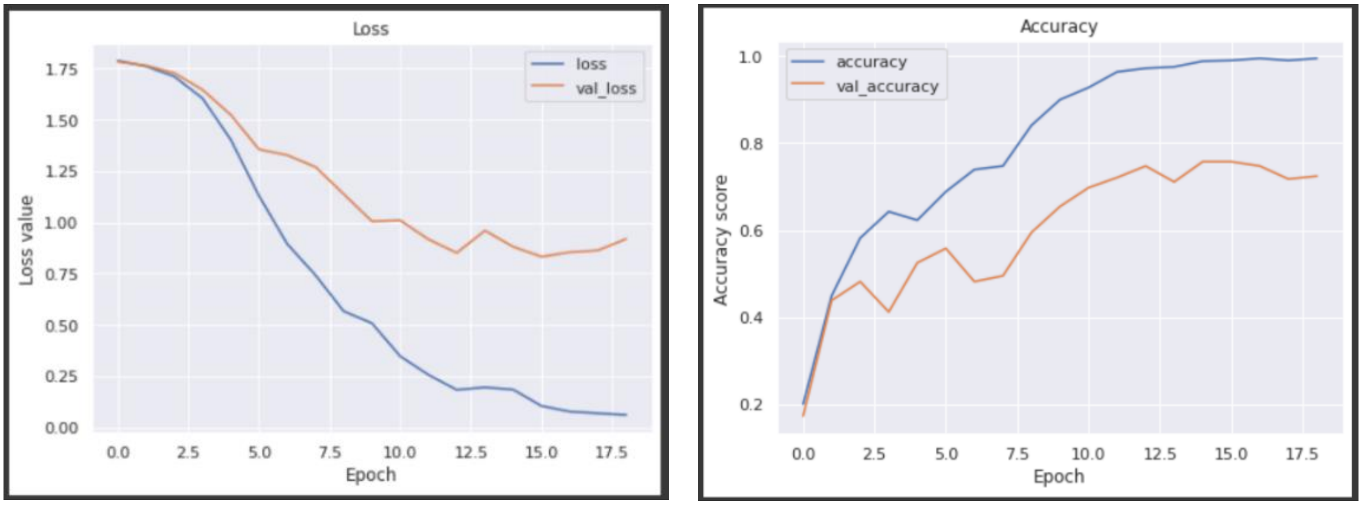
TABLE X
RESULT CONFUSION MATRIX LSTM AND GLOVE

	happiness	sadness	fear	disgust	anger	surprise
happiness	40	9	2	4	8	11
sadness	8	47	4	4	9	10
fear	6	3	40	4	5	4
disgust	5	2	2	39	2	0
anger	3	8	0	4	34	7
surprise	9	6	4	5	8	35

Finally, in the word embedding fast text architecture, the best architecture is obtained with 50 units and a dropout of 50, resulting in an accuracy of 0.731458. In the fast text training process, the relationship between loss and loss validation can be seen in Fig. 11, while the relationship for accuracy and accuracy validation can be seen in Fig. 12. Meanwhile, the configuration matrix test results on the fast text architecture can be seen in table XI.

TABLE XI
RESULT CONFUSION MATRIX LSTM AND FAST TEXT

	happiness	sadness	fear	disgust	anger	surprise
happiness	62	4	1	0	1	6
sadness	16	55	0	1	6	4
fear	3	0	54	2	3	0
disgust	1	1	0	47	1	0
anger	4	10	1	1	38	2
surprise	12	17	0	1	7	30



(a) Loss And Loss Validation LSTM And Fast Text

(b) Accuracy And Validation Accuracy LSTM And Fast Text

Fig. 9. The Fast Text Training Process

IV. CONCLUSION

In this study, the best accuracy, precision, recall, and f-1score results were in the following LSTM. Word embedding architecture, LSTM with Word2Vec with 50 units and 50 dropouts resulted in 73,15%. In comparison, LSTM with GloVe architecture with 50 units and 30 dropouts resulted in 60,1%, for LSTM with Fast Text architecture obtained the best architecture of 50 units and 50 dropouts, resulting in 73,15%. LSTM obtains the best accuracy with Word2Vec and LSTM with FastText, Fast Text has the advantage that it can handle the problem of out vocabulary, but in this study, it cannot improve the accuracy of Word2Vec. This study has not been able to produce very good accuracy; this is because of the data used. In future works, To get even better results, it is expected to apply other deep learning methods, such as CNN, BiLSTM, etc. It is hoped that more data will be used in future studies.

REFERENCES

[1] Ardiada, D., Sudarma, M., & Giriantari, D. (2019). *Text Mining pada Sosial Media untuk Mendeteksi Emosi Pengguna Menggunakan Metode Support Vector Machine dan K-Nearest Neighbour*. *18*(1), 55–60.

[2] Bata, J., Suyoto, & Pranowo. (2015). Leksikon Untuk Deteksi Emosi Dari Teks Bahasa Indonesia. *Seminar Nasional Informatika 2015 (SemnasIF 2015)*, *2015*(November), 195–202.

[3] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). *Enriching Word Vectors with Subword Information*. http://www.isthe.com/chongo/tech/comp/fnv

[4] Consoli, D. (2009). Emotions That Influence Purchase Decisions And Their Electronic Processing. *Annales Universitatis Apulensis Series Oeconomica*, *2*(11), 1–45.

[5] Dandannavar, P. S., Mangalwede, S. R., & Kulkarni, P. M. (2018). Social Media Text - A Source for Personality Prediction. *Proceedings of the International Conference on Computational Techniques, Electronics and Mechanical Systems, CTEMS 2018*, 62–65. https://doi.org/10.1109/CTEMS.2018.8769304

[6] Daouas, T., & Lejmi, H. (2018). Emotions recognition in an intelligent e-learning environment. *Interactive Learning Environments*, *26*(8), 991–1009. https://doi.org/10.1080/10494820.2018.1427114

[7] Dwi L, Adinda., Harijanto, Budi., Rahutomo, Faisal. (2020). *Implementasi Deep Learning Untuk Deteksi Ekspresi Emosi pada Twitter*.

[8] Fanesya, F., Wihandika, R. C., & Indriati. (2019). *Deteksi Emosi Pada Twitter Menggunakan Metode Naïve Bayes Dan Kombinasi Fitur*. *3*(7), 6678–6686.

[9] Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation. In *Part of the Computer Sciences Commons Comments*. https://digitalcommons.utep.edu/cs_techrephttps://digitalcommons.utep.edu/cs_techrep/1209

[10] Haddi, E., Liu, X., & Shi, Y. (2013). The role of text pre-processing in sentiment analysis. *Procedia Computer Science*, *17*, 26–32. https://doi.org/10.1016/j.procs.2013.05.005

[11] Halim, K., Novianus Palit, H., & Tjondrowiguno, A. N. (2020). Penerapan Recurrent Neural Network untuk Pembuatan Ringkasan Ekstraktif Otomatis pada Berita Berbahasa Indonesia. *Jurnal Infra*, *8*(1), 221–227.

[12] Han, J., & Kamber, M. (n.d.). *Data Mining: Concepts and Techniques*.

[13] Haryadi, D., & Kusuma, G. P. (2019). Emotion detection in text using nested Long Short-Term Memory. *International*

Journal of Advanced Computer Science and Applications, *10*(6), 351–357. https://doi.org/10.14569/ijacsa.2019.0100645

[14] Hochreiter, Sepp dan Schmidhber, Jurgen. (1997). Long Short-Term Memory. https://doi.org/10.1162/neco.1997.9.8.1735

[15] Juwiantho, H., Setiawan, E. I., Santoso, J., & Purnomo, M. H. (2020). Sentiment Analysis Twitter Bahasa Indonesia Berbasis Word2Vec Menggunakan Deep Convolutional Neural Network. *Jurnal Teknologi Informasi Dan Ilmu Komputer (JTIIK)*, *7*(1), 181–188. https://doi.org/10.25126/jtiik.202071758

[16] Lim, E., Istts, T. I., Setiawan, E. I., & Istts, T. I. (2020). *Stance Classification Post Kesehatan di Media Sosial Dengan FastText Embedding dan Deep Learning*. 65–73.

[17] Miedema, F. (2018). *Sentiment Analysis with Long Short-Term Memory networks*.

[18] Nazief, B. A. A. & Adriani, M. (2005). Confix- stripping: Approach to Stemming Algorithm for Bahasa Indonesia. *Conferences in Research and Practice in Information Technology Series*, *38*(4).

[19] Nurlaila, A., Wiranto, & Saptono, R. (2017). Classification Of Customers Emotion Using Naive Bayes Classifier (Case Study : Natasha Skin Care). *Jurnal Ilmiah Teknologi Dan Informasi*, *6*(2), 92–97.

[20] Nurrohmat, M. A., & SN, A. (2019). Sentiment Analysis of Novel Review Using Long Short-Term Memory Method. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, *13*(3), 209. https://doi.org/10.22146/ijccs.41236

[21] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, *June 2018*, 1532–1543. https://doi.org/10.3115/v1/d14-1162

[22] Rao, A., & Spasojevic, N. (2016). *Actionable and Political Text Classification using Word Embeddings and LSTM*. http://arxiv.org/abs/1607.02501

[23] Rohman, A. N., Handayani, R. D., & Kusrini, K. (2020). Deteksi Emosi Media Sosial Menggunakan Term Frequency-Inverse Document Frequency. *CSRID (Computer Science Research and Its Development Journal)*, *11*(3), 140–148.

[24] Rohman, A. N., Utami, E., & Raharjo, S. (2019). *View of Deteksi Kondisi Emosi pada Media Sosial Menggunakan Pendekatan Leksikon dan Natural Language Processing*. Jurnal Eksplora Informatika. https://doi.org/https://doi.org/10.30864/eksplora.v9i1.277

[25] S. B. Kotsiantis, D. K. and P. E. P. (2006). *Data Preprocessing for Supervised Learning*.

[26] Saifulloh, M., & Siregar, M. U. (2019). Pengungkapan Diri Gofar Hilman Sebagai Influencer Melalui Media Instagram. *Pustaka Komunikasi*, *2*(2), 167–180. https://doi.org/https://doi.org/10.32509/pustakom.v2i2.869

[27] Salam, A., Zeniarja, J., & Khasanah, R. S. U. (2018). Analisis Sentimen Data Komentar Sosial Media Facebook Dengan K-Nearest Neighbor (Studi Kasus Pada Akun Jasa Ekspedisi Barang J&T Ekpress Indonesia). *Prosiding SINTAK*, 480–486.

[28] Saputri, M. S., Mahendra, R., & Adriani, M. (2019). Emotion Classification on Indonesian Twitter Dataset. *Proceedings of the 2018 International Conference on Asian Language Processing, IALP 2018*, *November*, 90–95. https://doi.org/10.1109/IALP.2018.8629262

[29] Utomo, B. (2020). *Kinerja Deep Learning dalam Analisis Sentimen*.

[30] Arliyanti Nurdin, Bernadus Anggo Seno Aji, Anugrayani Bustamin, Z. A. (2020). Perbandingan kinerja Word Embedding Word2Vec, GloVe dan FastText pada klasifikasi teks. Jurnal TEKNOKOMPAK, 14(2), 74--79.

[31] Ekman (1999). Basic Emotions. In Encyclopedia of Personality and Individual Differences (pp. 1–6). https://doi.org/10.1007/978-3-319-28099-8_495-1