Journal of Engineering Science and Technology
Vol. 15, No. 5 (2020) 3242 - 3256

SENTIMENT ANALYSIS USING RECURRENT NEURAL NETWORK-LSTM IN BAHASA INDONESIA

LILIS KURNIASARI[1, 2, *], ARIEF SETYANTO[2]

[1]Magister Student of Department. Informatics Engineering, Postgraduate Program, Universitas Amikom Yogyakarta, Yogyakarta, Indonesia
[2]Department Informatics Engineering, Postgraduate Program, Universitas Amikom Yogyakarta, Department. Information Technology, Faculty of Engineering, Universitas Muhammadiyah Yogyakarta, Indonesia
*Corresponding Author: rainforest02@gmail.com

Abstract

The opinion of different individuals is an essential factor that needs to be considered in decision making. Due to the current competitive business environment, understanding customer opinions determine the success of modern companies. Various machine learning algorithms are used in Automatics opinion polarization mining from online text sources such as social media, user comments, and reviews. This approach has been used in many languages, including Indonesian. The purpose of this study therefore is to proposes the Recurrent Neural Network (RNN)-Long Short Memory Term (LSTM) in classifying sentiment polarity of Indonesian sentences. It evaluates the proposed algorithm with a dataset from traveling site reviews consisting of 25,000 reports in two classes of equal proportion (positive and negative). According to the evaluation results, the model has 95.0% accuracy.

Keywords: Deep learning, Long short memory term, Neural network, Recurrent Neural Network, Sentiment analysis.

1. Introduction

Natural language processing (NLP) has been one of the most active research areas for two decades. Its popularity attracts researchers to examine data, web, and text mining's, and information retrieval [1]. NLP has expanded from computer to management and social sciences, benefitting related fields such as marketing, finance, politics, communication, and history significantly. Opinion plays a vital role in almost all human activities and affects behaviour. It drives beliefs, perceptions of reality, choices, and to a certain extent, impacts the way others see and evaluate the world. Individuals and organizations need third opinions in decision making [2-5]. Therefore, organizations need to listen to users' opinions on their products and services. Furthermore, there are several ways of analysing opinions, including sentiment analysis. These are systems that automatically identify and process information, such as learning techniques [2, 6]. This study examines how deep learning is implemented in sentiment analysis in Bahasa. It aims to help organizations understand their customer opinion and perceptions of the product.

There are different forms of machine learning, including structured learning, which is deep and hierarchical. For each problem and learning technology, there are various deep learning architectures that can be used [7, 8]. Each architecture is built on a different network from its respective functions, such as a recurrent neural network (RNN). NLP has a temporal aspect, where a term in the sentence depends on the word before and after it. This research used RNN to reduce the dependence of words in a sentence. RNN associates each word with a certain time step in the order of input sentences. Each number of time steps is supposed to be equal to the maximum sequence length of each word.

There is a new component referred to as a hidden vector which summarizes all information in the previous time steps. Traditionally, a simple strategy for sequence modelling is used to map the order of inputs to fixed-size vectors using one RNN [9]. Additionally, the Long Short-Term module (LSTM) was also implemented to help overcome gradient vector growth components that potentially exist in the RNN algorithm during the training [10]. Gradient vector growth causes the RNN network to experience difficulties in learning nested sequences [10]. Sentence representation is quite difficult in natural language processing, including sentiment analysis. LSTM has superior performance in sentence modelling that deals with such challenges [11].

The purpose of this research is to evaluate and apply deep learning models, specifically, RNN-LSTM. This is achieved using the user reviews dataset from the Traveloka website, which receives 28.92 million visitors and 78.49% user traffic in Indonesia [12, 13]. In 2019, it had more than 500,000 hotel user reviews in Bahasa. This research considers positive and negative opinion polarization in training and testing the model. Most of the existing work on natural language processing (NLP), including sentiment analysis, uses English text [14, 15]. RNN - LSTM had excellent performance in several studies on sentiment analysis using the English text dataset [16, 17]. Currently, the human labelled sentiment dataset in Bahasa rarely available. Dataset twitter and Instagram consist of only less than ten thousand sentences [18, 19]. This study furthermore evaluates the RNN - LSTM to recognize a sentence sentiment polarization in Indonesian text. It also develops a sentiment dataset in Bahasa, which has 25,000 sentences, both positive and negative.

2. Literature Review

There are several studies on sentiment analysis using various algorithms, including Naive Bayes. Liu et al. [20] examined the scalability of Naive Bayes classifier on big data to achieve fine-grain control from the analysis procedure in the movie review dataset. Go et al. [14] used three algorithms, including Naive Bayes, maximum entropy, and support vector machine to classify twitter messages using distant supervision. Some research uses deep learning and neural networks to analyse sentiments, though most of them use English and Asian languages [15, 21-25].

Thomas and Latha [16] used a recurrent neural network to analyse sentiment from tweeters in southern Malayalam language. The results showed that the accuracy of the model made using the RNN-LSTM method was 80%. The accuracy score can still be improved using a deeper dataset. The most outstanding aspect of this research is that this method can be implemented using other languages. The RNN-LSTM method is the model used to analyse sentiment.

Miedema and Bhulai [17] studied why the RNN-LSTM model works well in analysing sentiments and how the models perform. LSTM is used to classify sentiments with a movie review dataset. The results show that the model can correctly classify 86.74% of the total reviews into the validation set. This model is quite sensitive to overfitting but provides excellent results even without parameter setting.

Wang et al. [26] stated that traditional RNN is not strong enough to handle complex sentiment expressions, and therefore, LSTM is implemented to classify sentiments. An experiment was conducted with a corpus of tweeters, a dataset containing 800,000 tweets labelled positive and negative. The results show that the LSTM network outperforms all other methods, including SVM and Naïve Bayes.

A few research has been conducted on sentiments analysis in Indonesian text using machine learning techniques. Watrianthos et al. [12] used a dataset from the Traveloka user review on the play store. This study aims to determine user perceptions based on service quality measurements using the Naive Bayes method. Iswanto and Poerwoto [27] used Naïve Bayes classifiers, Maximum Entropy classifiers, and Support Vector Machines in pre-processing the twitter dataset. This method achieves the recall and precision of up to 85.50%. Kurniawan et al. [28] research using a 1.720 datasets in Bahasa had an average F measure of 75.18%. Hermanto et al. [18] used a Naïve Bayes method for classifying twitter user reviews in Bahasa. This research classified reviews in positive and negative aspects. The details are summarized in Table 1.

Table 1. Literature review resume.

Reference	Dataset positive/Negative/neutral	Lan-guage	Classifier	Acc (%)
Shirani-Mehr [21]		English	CNN + word2ve	46
Liu et al. [20]	1000/1000	English	Naïve Bayes	82
Go et al. [14]	800k/800k	English	SVM	81
Timmaraju and Khanna [15]	5000/5000	English	SMV Linier 2-layer NN RecNN-RNN	86.49 83.94 83.88
Al-smadi et.al. [22]	24028	Arabic	SVM K-Nearest Neighbor	95.40 94.10
Zhang and Chen [23]	929/946	Chinese	CNNs	88.7
Heikal et al. [24]	2500/2500	Arabic	CNN + LSTM	64.46

Reference	Dataset positive/Negative/neutral	Lan-guage	Classifier	Acc (%)
Pasupa and Na Ayutthaya [25]	309/298/508	Thai	CNN LSTM Bi-LSTM	71.40 60.20 60.50
Thomas and Latha [16]	2500/2500	Malayalam	RNN + LSTM	80
Miedema and Bhulai [17]	50000/50000	English	RNN + LSTM	86.7
Watrianthos et al. [12]		Bahasa	Naïve Bayes	31.02
Iswanto and Poerwoto [27]		Bahasa	SVM	85.50
Kurniawan et al. [28]	430/430/860	Bahasa	Naïve Bayes	75.18
Hermanto et al. [18]		Bahasa	Naïve Bayes	

3.Methodology

Figure 1 illustrates the proposed framework. A total of 1,8 million reviews were collected from the Traveloka website during 2018 august. The data is used as a corpus to create word vectors. Positive and negative 25000 reviews and labels were randomly selected. The labelled reviews are used as training and testing data sets. Word2vec is a two-layer neural network that processes words in the corpus into a collection of vectors for the inner network to understand them. Our model uses an RNN-LSTM algorithm to classify user reviews. Word2vec transformation needs to transform the word into a vector to satisfy the required input of the LSTM network.

Apart from creating word vectors, word2vec group vectors from words with definitions and uses in the same context into vector space. Word2vec detects every word vector mathematically. Words with the same distance are often close to each other in vector space. Representation of word vectors is called embedding words. Fairly accurate guesses can be made on the meaning of the word with enough data and context. This improves the performance of the recurrent neural network algorithm used in the model.

3.1. Word2Vec

Google developed word2vec initially implemented two algorithms called skip-gram models and a continuous bag of the word (CBOW) [29]. Word2vec takes a word from the corpus as input and produce word vector as output. It works optimally in guessing words accurately if there are sufficient data, usefulness, and context of each word.

We trained the word2vec model using 1,8 million review data from the Traveloka website with 138280 words. Words often used in the same context are in a similar group in vector space. Grouping words by context help the model to reduce error rates. The model produces a word vector, each representing the word and its context. Besides, it also represents the word meaning and semantics [30].

Figure 2 shows an example of the implementation of the word2vec model in Bahasa and English. On the left, a piece of Word2vec result in Bahasa, it group bersih (clean) and kotor (dirty) words. Semantically, closer words are located firmly in the vector space on the right side of a word2vec for English. Table 2 shows the word vector in ‘Bahasa’ and ‘English’. ‘bersih’ is adjacent to the word vector ‘bersih’, ‘bersih’, ‘bersih’, ‘bersi’, and clean close to bright, shining, fresh, while dirty close to dusty, messy, and muddy. RNN requires this cluster for the starting point of training.

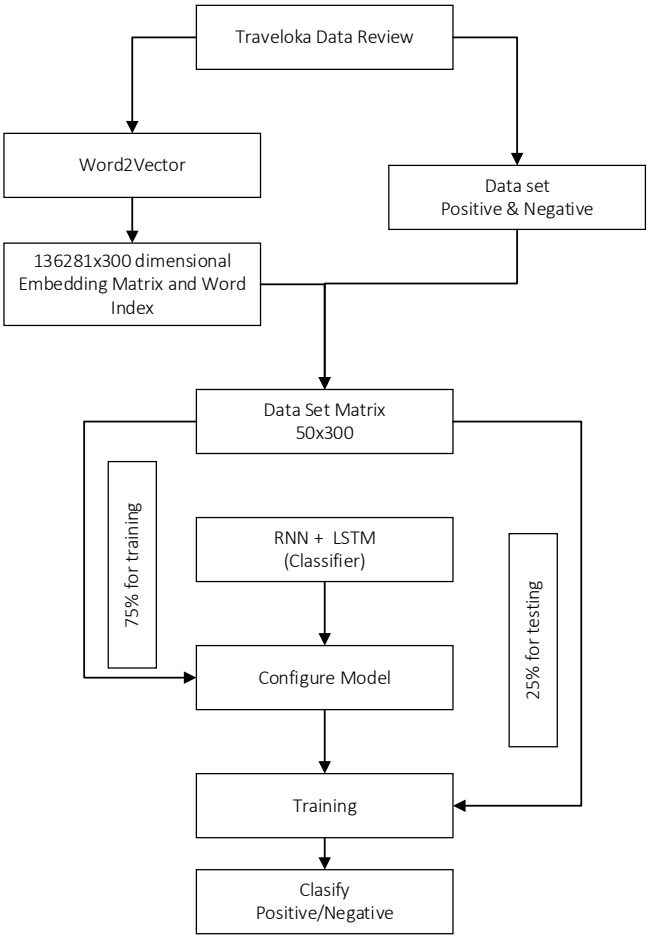

Fig. 1. The proposed framework.

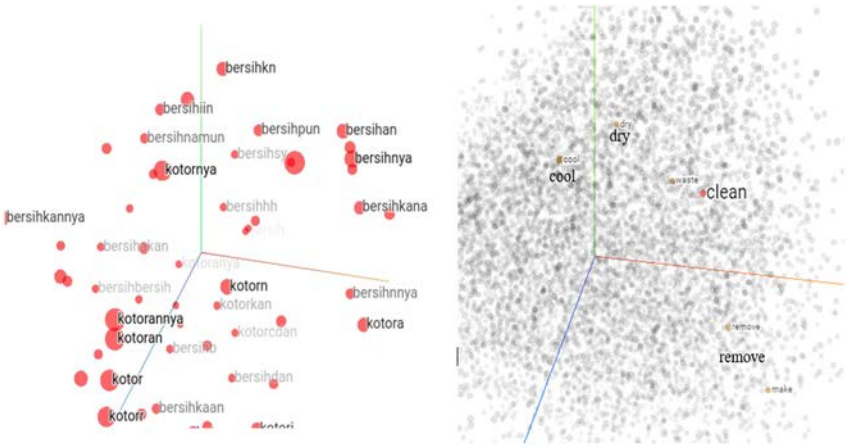

Fig. 2. Word with same context (left in Bahasa, right in English).

Table 2. Words vector.

Bahasa	**Word**	*Bersih*	*bersihkan*	*Kotor*	*kotoran*
	Vector	0.380360	0.379824	0.527863	0.460976
English	**Word**	Clean	cleaning	dirty	dirt
	Vector	0.658972	0.678098	0.689567	0.690235

Word vector model produces a matrix with 136,281 words vectors, each with a dimension of 300. Two data structures were created for the training process. Firstly, 136,281 words vectors were made to a list in Python. Secondly, 136,281 x 300 embedding matrices to load all values of the word vectors were made.

The next step involves making a vector representation of the input sentence. The function of Tensorflow embedding was utilized to get word vectors. Embedding tensor has a function for retrieving the embedding matrix. It is also useful for retrieving the index of each word in the sentence. The index of each word also represents the index of the word vector. This is basically a row index of each word in the input sentence. Table 3 shows an example index word vector for a sentence "pagi pagi sarapan nasi goreng di Kasur sambil nonton renang".

Table 3. Index words vector.

sentence	*Pagi Pagi sarapan nasi goreng di Kasur sambil nonton renang*								
word	*Pagi*	*Pagi*	*Sarapan*	*Nasi*	*Goreng*	*Di*	*Kasur*	*Sambil*	*Nonton*
index	548	548	130	225	226	19	461	1068	2669

3.2. Recurrent neural network

Recurrent neural network architecture overcomes temporal aspects in NLP. This is where the word in the sentence depends on the word before or after it. In the RNN structure, each word in the sentence is connected to a certain time in sequence according to the input. Therefore, the number of steps is the same maximum length of the word sequence in the sentence, as shown in Table 4.

Table 4. Number of time step.

Sentence	Pagi	Pagi	Sarapan	Nonton
Sequence length	X_0	X_1	X_3	X_9
Time	t =0	t =2	t =3	t =4

RNN applies current and past input sources for the process. Fig. 3 explains the sentence "pagi pagi sarapan nasi goreng di Kamar" as an input at the moment and CONTEXT UNIT as the output of the previous moment or input in the past. The last moment output in time step or t-1 has an impact on the current moment's output at the time step or *t*.

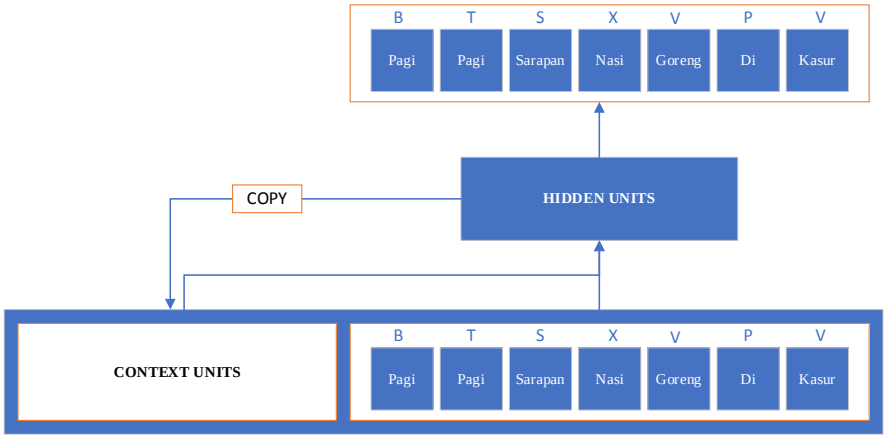

Fig. 3. Input RNN.

However, the RNN maintains this sequential information in a hidden state. This process achieves many time steps to impact the new process. It is formulated into a mathematical Eq. (1), where *ht* hidden state is a function of the current input, multiplied by the weight matrix *w*, then added input to the past *ht-1* multiplied by the transition matrix *u*. Weight matrices determine adjustments to current input and the previously hidden conditions. Errors that occur are reprocessed by backpropagation until the lowest error rate is reached.

$$h_t = \sigma(w^H h_{t-1} + w^x x_t) \quad (1)$$

To update the weight matrix, Adam optimization is used to process backpropagation through time. Afterward, the hidden state vector in the last time step is entered into the Binary Softmax classifier, which produces values of 0 and 1, or provides the possibility of positive or negative sentiment as shown in Fig. 4. Tensorflow with CUDA was used in the training model. The device supporting the training process uses two GPUs from Invidia 1070 ti. KERAS is used in the deep learning process.

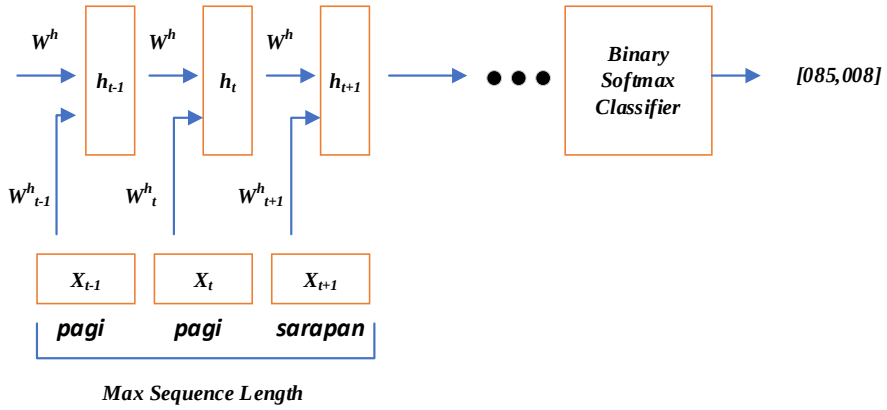

Fig. 4. Binary SoftMax classifier.

3.3. Long short-term memory units (LSTM)

Long Short-Term Memory is a module in an RNN network that solves missing gradient problems. In general, RNN applies the LSTM network to avoid propagation errors. This enables the RNN learning through many steps of time. LSTM contains cells that store information outside a recurring network. The cell is like the memory in the computer, deciding when the data needs to be stored, written, read, or deleted through the gate, as shown in Fig. 5. There are four gates that LSTM use, including an input, forget, and output gates, and a new memory container [30].

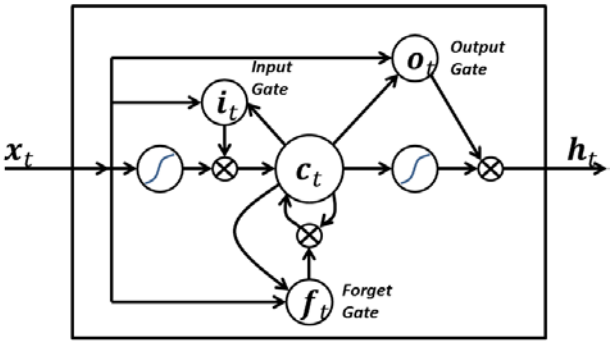

Fig. 5. LSTM gate.

The study uses four hyper parameters to improve the performance of RNN and LSTM effectively. The learning rate starts from 0.001, and it is used by RNN to maintain fluctuations and training processes. Adam optimizer is utilized since it has properties with adaptive learning levels. Also, the 64 LSTM units are used for this model training, though the number of units depends on the average length of each review. Finally, from the previous embedding matrix process, the size of the word vector 136281×300 is used.

4. Results and Discussion

4.1. Data set

This study used the Traveloka travel web and mobile application review dataset. It consisted of twenty-five thousand datasets divided into positive and negative classes, each with 12,500. The dataset is stored in a CSV file. Table 5 shows the review dataset.

Table 5. Review data set.

	Positive review	Negative review
Total File	12.500	12.500
Total words	2.989.896	3.353.227
Average words	28.99	33.53

Figure 6 shows a histogram of the sentence length distribution of the review dataset. Matplot library was used to visualize the data. We apply the visualization histogram to know the average number of words each review. We use the average

word value as the value of the max sequence length of the model. Figure 6 shows the distribution of sentence length distributed between 20 to 140 words. According to the histogram, the average review is under 50 words. The average value is used as the max sequence length value at 50. Therefore, an input of over 50 words is truncated.

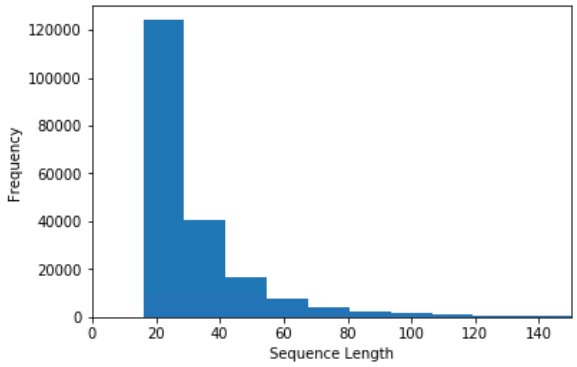

Fig. 6. Histogram review of data set.

4.2. Experimental data set matrix

The model used four hyper parameters to improve the performance, as shown in Table 6. In this work, there is a need to specify placeholders. Two placeholders were created, one as input into the network and the other as labels.

Table 6. Hyper parameters.

No.	Hyper parameter	Total
1	Batch size	24
2	LSTM unit	64
3	Number of classes	2
4	Number of training iteration	10000

Labels' placeholders contain a set of [1.0] values representing a positive review dataset and [0.1] representing a negative review dataset. Placeholders have rows that represent the input of each training dataset. The model also uses placeholders to hold input data as input into the system, as shown in Fig. 7.

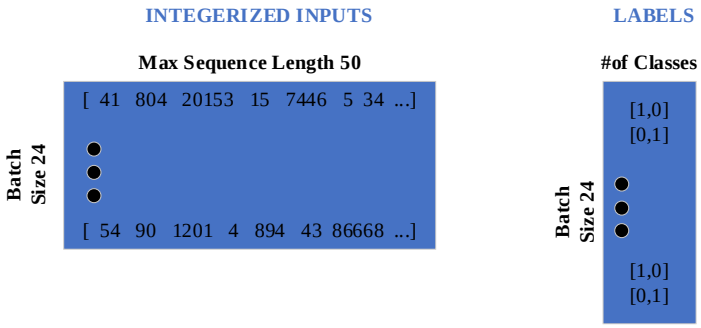

Fig. 7. Placeholder on LSTM.

With placeholders, the model runs the TensorFlow lookup function to capture word vectors generated by word2vec in the previous process. The word vectors, also called embedding matrices, with placeholders return the 3-D batch tensor dimensions with the maximum dimension sequence length of the embedding matrix shown in Fig. 8. Using the 3-D test makes it easier to visualize the input data in an integer form on TensorFlow.

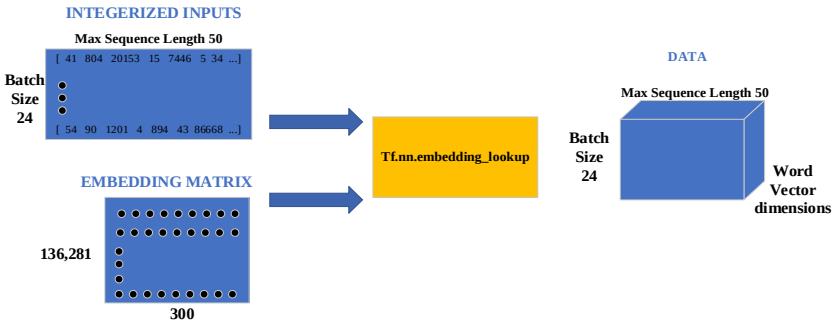

Fig. 8. 3-D dimension.

The results of the above process form data as integers and input into the LSTM network. The integer data is placed into the LSTM unit used with a function in Jupiter. After the LSTM is filled with data, the system wraps the cell in the dropout layer to prevent overfitting. The selection of LSTM network architecture helps to utilize hidden state vectors. This process is achieved by stacking a lot of LSTM cells. This ensures the model maintains information from long-term dependence; otherwise, it can be used to provide parameters to the model. Therefore, the LSTM network usually takes a little longer in the learning process and more training examples. Figure 9 shows an overview of the model. The input is an embedding matrix with a max sequence length of 50. Afterward, 64 LSTM cell with four gates each was used. The output layer using binary SoftMax process 64 LSTM outputs to make predictions [1.0] or [0.1].

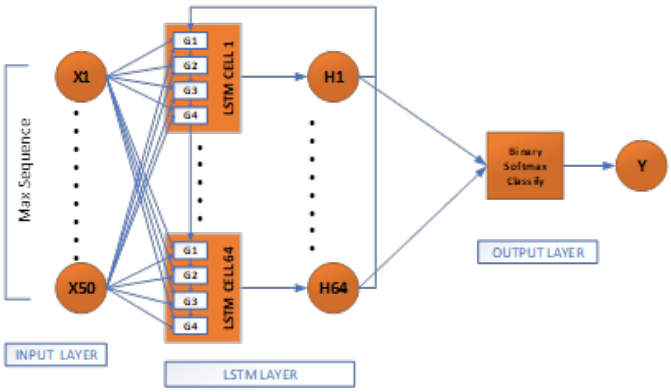

Fig. 9. LSTM network.

The first output of the dynamic RNN function can be considered as the last hidden state vector. This vector might be reshaped and multiplied by a final weight

matrix and a bias term to obtain the final output values [25]. The model ensures the prediction formulation works correctly using the maximum value index of all two output values matched with the training data set label.

4.3. Result and analysis

The first analysis step involves defining Tensorflow and loading the reviews and labels, then running function in the training session. This function has two arguments, specifically fetches and feed_dict. Essentially, the fetches argument defines the value of the computational process, while the feed_dict argument enters the data. Both arguments are for floating all input from the placeholders. The data structure serves as input to batch reviews and labels. Also, the training process repeats according to the number of iterations, which are set to 10000 iterations. From the training set, an accuracy of 0.950% and the loss value of close to zero% was obtained.

Figures 11 and 10, the losses decrease while performance accuracy increases. However, it should be monitored for possible overfitting during training, which is a common phenomenon in machine learning [27]. It occurs when the model becomes compatible with the dataset to reduce losses to zero. Therefore, training stops before the loss value reaches zero to anticipate overfitting based on intuition techniques. For comparison, training using other machine learning algorithms such as CNN, Naive Bayes, and conventional RNN was conducted.

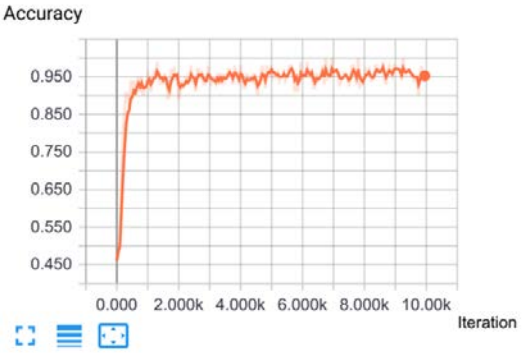

Fig. 10. Model training accuracy.

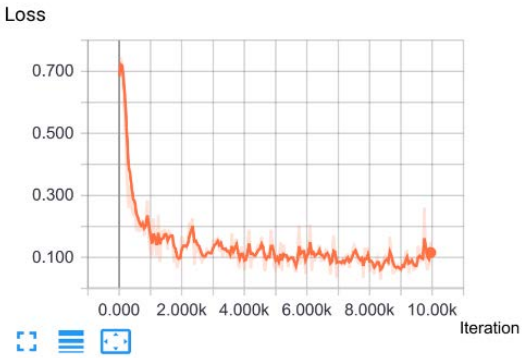

Fig. 11. Model training loss.

The evaluation result of each algorithm is shown in Table 7. These results indicate that RNN-LSTM achieves better accuracy compared to the other model. The accuracy cannot be directly compared since the dataset is different.

Table 7. Result comparison.

Model	accuracy	dataset
CNN + Word2Vec [31]	0.91323	UCI & Kaggle
Naïve Bayes [31]	0.441	UCI & Kaggle
RNN Conv [32]	0.8977	WWW.JD.COM
RNN + LSTM	0.950	www.traveloka.com

A model validation test was conducted using four-fold cross-validations. The dataset was divided into four equal sizes, as shown in Table 8. Furthermore, four experiments were conducted with composition, as shown in Table 9.

The validation test results are shown in Table 10. From the validation test, the average value of accuracy of 94.88% and a standard deviation of 0.4426 were obtained.

Table 8. Data set group.

Fold 1	Fold 2	Fold 3	Fold 4
25%	25%	25%	25%

Table 9. Experiment cross-validation.

Experiment	Testing			Training
exp 1	Fold 1	Fold 2	Fold 3	Fold 4
exp 2	Fold 1	Fold 2	Fold 4	Fold 3
exp 3	Fold 1	Fold 3	Fold 4	Fold 2
exp 4	Fold 2	Fold 3	Fold 4	Fold 1

Table 10. Confusion matrix.

Exp	Accuracy	Precision	Recall	F1 Score
1	94.44%	90.21%	97.81%	93.64%
2	96.11%	96.78%	93.97%	95.73%
3	93.74%	94.45%	89.98%	92.45%
4	95.24%	92.87%	96.51%	94.89%

5. Conclusions

This study proposed a model for sentiment analysis using RNN-LSTM for the Indonesian language dataset with 25,000 human labelled sentences. The RNN in the model is implemented using the TensorFlow framework which helps to classify sentiments and understand the natural languages. Moreover, the results from the experiments showed that RNN-LSTM has 95.0% accuracy, outperforming CNN+word2vec, naïve Bayes, and RNN Conv. In the training process, intuition techniques were used to solve overfitting problem that often occurs during the training process. However, scientists have recently proposed grid LSTM with high performance in some applications, though it should be used to improve the

performance of the sentiment classifier in future. This research used only positive and negative data sets. Future projects should use the three label data sets with mine neutral to evaluate the performance of the model.

Nomenclatures

C_t	New memory container
F_t	Forget gate
ht	Hidden state of time step
h_{t-1}	Hidden state of the previous time step
i_t	Input gate
O_t	Output gate
W^H	Matrix-based on previous hidden state
W^X	Matrix-based on the current input
X_t	Time step

Greek Symbols

σ	Tanh.

Abbreviations

CNN	Convolution Neural Network
CSV	Comma-Separated Values
LSTM	Long-Short Term Memory
NLP	Natural Language Processing
RNN	Recurrent Neural Network
SVM	Support Vector Machine

References

1. Rezaeinia, S.M.; Rahmani, R.; Ghodsi, A.; and Veisi, H. (2019). Sentiment analysis based on improved pre-trained word embeddings. *Expert Systems with Applications*, 117, 139-147.
2. Zhang, L.; Wang, S.; and Liu, B. (2018). Deep Learning for Sentiment Analysis : A Survey. *CoRR*, abs/1801.0, 32.
3. Appel, O.; Chiclana, F.; Carter, J.; and Fujita, H. (2017). A Consensus Approach to the Sentiment Analysis Problem Driven by Support-Based IOWA Majority. *International Journal of Intelligent Systems*, 32(9), 947-965.
4. Appel, O.; Chiclana, F.; Carter, J.; and Fujita, H. (2017). Cross-ratio uninorms as an effective aggregation mechanism in sentiment analysis. *Knowledge-Based Systems*, 124, 16-22.
5. Appel, O.; Chiclana, F.; Carter, J.; and Fujita, H. (2018). Successes and challenges in developing a hybrid approach to sentiment analysis. *Applied Intelligence*, 48, 1176-1188.
6. Cambria, E. (2016). Affective Computing and Sentiment Analysis. *IEEE Intelligent Systems*, 31(2), 102-107.
7. Song, M.; Park, H.; and Shin, K. shik. (2019). Attention-based long short-term memory network using sentiment lexicon embedding for aspect-level sentiment analysis in Korean. *Information Processing and Management*, 56(3), 637-653.

8. Dosoula, N.; Griep, R.; and Ridder, R.D. (2016). Sentiment analysis of multiple implicit features per sentence in consumer review data. *Frontiers in Artificial Intelligence and Applications*, 291, 241-254.
9. Cho, K.; Merrienboer, B.V.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* 1724-1734.
10. Kolen, J.F.; and Kremer, S.C. (2009). *A Field Guide to Dynamical Recurrent Networks*. Wiley-Institute of Electrical and Electronics Engineers Press.
11. Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735-1780.
12. Watrianthos, R.; Suryadi, S.; Irmayani, D.; Nasution, M.; and Simanjorang, E.F.S. (2019). Sentiment Analysis Of Traveloka App Using Naïve Bayes Classifier Method. *International Journal of Scientific & Technology Research*, 9(8), 786-788.
13. Website Performance (2019). worldwide-overview. Retrieved 15 November 2019, From https://pro.similarweb.com
14. Go, A.; Bhayani, R.; and Huang, L. (2009). Twitter Sentiment Classification using Distant Supervision. *CS224N Project Report*.
15. Timmaraju, A.; and Khanna, V. (2015). Sentiment Analysis on Movie Reviews using Recursive and Recurrent Neural Network Architectures. *Semantic Scholar*, 1-6.
16. Thomas, M.; and Latha, C.A. (2018). Sentimental analysis using recurrent neural network. International *Journal of Engineering & Technology*, 7, 88-92.
17. Miedema, F.; and Bhulai, S. (2018). *Sentiment Analysis with Long Short-Term Memory networks*. Research paper. Vrije Universiteit Amsterdam.
18. Hermanto, D.T.; Ziaurrahman, M.; Bianto, M.A.; and Setyanto, A. (2018). Twitter Social Media Sentiment Analysis in Tourist Destinations Using Algorithms Naive Bayes Classifier. *Journal of Physics: Conference Series* 1140,
19. Rosanensi, M.; Madani, M.; Wanggono, R.T.P.; Setyanto, A.; and Wahyuni, S.N. (2018). Analysis sentiment and tourist response to rinjani mountain tour based on comments from photo upload in instagram. in *Proceedings - 2018 3rd International Conference on Information Technology, Information Systems and Electrical Engineering*, 184-188.
20. Liu, B.; Blasch, E.; Chen, Y.; Shen, D.; and Chen, G. (2013). Scalable sentiment classification for Big Data analysis using Naïve Bayes Classifier. *IEEE International Conference on Big Data, California USA*, 99-104.
21. Shirani-Mehr, H. (2015). *Applications of Deep Learning to Sentiment Analysis of Movie Reviews*. research paper. Department of Management Science & Engineering Stanford University.
22. Al-smadi, M.; Al-ayyoub, M.; Jararweh, Y.; and Qawasmeh, O. (2019). Enhancing Aspect-Based Sentiment Analysis of Arabic Hotels ' reviews using morphological , syntactic and semantic features. *Information Processing and Management*, 56(2), 308-319.
23. Zhang, L.; and Chen, C. (2016). Sentiment Classification with Convolutional Neural Networks: An Experimental Study on a Large-Scale Chinese

Conversation Corpus. *12th International Conference on Computational Intelligence and Security*, 165-169.

24. Heikal, M.; Torki, M.; and El-Makky, N. (2018). Sentiment Analysis of Arabic Tweets using Deep Learning. *Procedia Computer Science*, 142, 114-122.
25. Pasupa, K.; and Na Ayutthaya, T.S. (2019). Thai sentiment analysis with deep learning techniques: A comparative study based on word embedding, POS-tag, and sentic features. *Sustainable Cities and Society*, 50, 101615.
26. Wang, X.; Liu, Y.; SUN, C.; Wang, B.; and Wang, X. (Association for Computational Linguistics, 2015). Predicting Polarities of Tweets by Composing Word Embeddings with Long Short-Term Memory. in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Vol 1: Long Papers)* 1343-1353.
27. Iswanto, B.H.; and Poerwoto, V. (2018). Sentiment analysis on Bahasa Indonesia tweets using Unibigram models and machine learning techniques. in *IOP Conference Series: Materials Science and Engineering* 434, 434.
28. Kurniawan, S.; Kusumaningrum, R.; and Timu, M.E. (2019). Hierarchical Sentence Sentiment Analysis of Hotel Reviews Using the Naïve Bayes Classifier. *2018 2nd International Conference on Informatics and Computational Sciences, ICICoS 2018*, 104-108.
29. Rong, X. (2014). word2vec Parameter Learning Explained. *ArXiv*, *1411*..
30. Deshpande, A.(2018). Perform sentiment analysis with LSTMs, using TensorFlow - O'Reilly Media. Retrieved 20 April 2019, from https://www.oreilly.com/learning/perform-sentiment-analysis-with-lstms-using-tensorflow
31. Panthati, J.; Bhaskar, J.; and Ranga, T.K. (2018). Sentiment Analysis on Customer Reviews using Deep Learning. *International Journal of Computer Sciences and Engineering*, 6, 1023-1024.
32. Li, D.; and Qian, J. (2016). Text sentiment analysis based on long short-term memory. *2016 1st IEEE International Conference on Computer Communication and the Internet, ICCCI 2016*, 471-475.