Comparison of Accuracy between Long Short-Term Memory-Deep Learning and Multinomial Logistic Regression-Machine Learning in Sentiment Analysis on Twitter

Aries Muslim¹, Achmad Benny Mutiara², Rina Refianti³, Cut Maisyarah Karyati⁴, Galang Setiawan⁵
Faculty of Computer Science and Information Technology
Gunadarma University
Jl. Margonda Raya No.100, Depok 16424, Indonesia

Abstract—The paper is about sentiment analysis research on Twitter. In this research data with the keyword, ‘Russian Hacking’ concerning the 2016 US presidential election on Twitter was taken as a dataset using Twitter API with Python programming language. The first process in sentiment analysis is the cleaning phase of tweet data, then using the Lexicon-based method to produce positive, negative, and neutral sentiment values for each tweet. Data that has been cleaned and classified will be processed in the Deep learning method with Long Short-Term Memory (LSTM) algorithm and Machine learning method with Multinomial Logistic Regression (MLR) algorithm. The accuracy of these two classification methods are calculated using the confusion-matrix method. The accuracy obtained from the LSTM classification method is 93% and the MLR classification method is 92%. Thus, it can be concluded that LSTM is better in classifying sentiments compared to MLR.

Keywords—Sentiment analysis; deep learning; machine learning; Long Short-Term Memory (LSTM); Multinomial Logistic Regression (MLR)

I. INTRODUCTION

The demand for information technology needs in this era is increasing, especially in terms of communication. One form of technological progress is social media. Twitter is becoming the dominant form of social media. Twitter is a website that is a service of a microblog, which is a form of a blog that limits the size of each post, which provides facilities for users to be able to write messages in Twitter updates which contain only 140 characters [21], [14]. Twitter users can express various opinions and opinions. Forms of expressions written by users on twitter are called tweets. The number of large tweets shared by Twitter users every second, making the collection of tweets can be processed and analyzed to find out a review or public opinion about a particular product, service, or topic.

Sentiment analysis [1], [3], [19] is a research branch of text mining that is used to analyze and classify opinions from a text document. Sentiment analysis is the process of extracting, processing and understanding textual data to get information in the form of opinions or tendencies of opinion on a problem or object by someone, whether it tends to have a negative or positive opinion or opinion. Sentiment analysis can be done with many methods, one of which is the Machine learning and Deep learning methods [4].

Based on these reasons, the authors are interested in conducting sentiment analysis research. The research conducted is to make a comparison on the application of the Deep learning classification method using the Long Short-Term Memory (LSTM) algorithm and Machine learning classification method using Multinomial Logistic Regression (MLR) algorithm. The application of the classification method is done by using data taken from Twitter with the help of the Twitter API application and the Python programming language. The analyzed topic is the case of Russian hacking that concerns the US presidential election in 2016. Data are taken via Twitter based on Hastag(#) relating to research topics. The accuracy results obtained using the LSTM classification method will be compared with the MLR classification method, so that it can be known which algorithm is capable of producing classification methods with better accuracy values.

In the rest of paper, we show briefly the literature review and related work in Section II. In Section III the research methodology is presented. The implementation and results related to our research are also shown in Section IV. The last section is conclusion and future work of our research.

II. REVIEW OF LITERATURE

A. Sentiment Analysis

Sentiment analysis is a type of natural language processing to track people’s moods about certain products or topics. Sentiment analysis, which is also called opinion mining [17], involves building a system to collect and examine opinions about products or services made in web posts, blogs, or comments on social media. Sentiment analysis is useful in many ways. For example, in marketing [7], [10], it helps the success of new advertisements or product launches, determining which version of the product or service is popular and even identifying the types of demographics that like or do not like certain features.

The basic task in sentiment analysis is to classify the text that is in a sentence or document, then determined the opinions expressed in sentence or whether the document is positive, negative or neutral. As in [15], [16], sentiment analysis can also express emotional feelings of sadness, joy, or anger.
B. Lexicon-based Method

The Lexicon-based method can [16], [20]; i) identify the sentiments of each opinion words contained in the tweet data, and ii) handle multi-opinion problems in a data. This method is an improvement from the method that cannot handle multi-opinion problems. In handling the multi-opinion problem, this method collects all sentiments from the word opinion based on the distance between the words of opinion and its features. So that finally it can be used to determine the class of opinion of each data. In this method, the data is divided into three parameters of sentiment analysis, namely positive, negative, and neutral.

C. Long Short Term Memory (LSTM)

Long Short Term Memory (LSTM) is another type of processing module for Recurrent Neural Network (RNN). LSTM was created by Hochreiter & Schmidhuber (1997)[11] and later developed and popularized by many researchers [9], [8], [2], [5], [6], [13]. Like RNN, the LSTM network also consists of modules with repetitive processing. The difference is that the modules that make up the LSTM network are LSTM modules [18], as seen in Figure 1:

The diagram in Figure 3 explains that each line carries the entire vector, from the output of one node (node) to another input. The pink circle represents the operation of elements, such as the addition or multiplication of vector elements, while the yellow box is the neural network layer (containing parameters and biases) that can study. Two lines joining indicate a combination of two matrices or vectors, while the split line indicates the content is copied and the copy goes to a different node.

1) LSTM Key Mechanism: The key idea of LSTM is the path that connects the old context \( C_{t-1} \) to the new context \( C_t \) at the top of the LSTM module, as illustrated below in Figure 4.

The LSTM module (one green box) has different processing from the regular RNN module. Another difference is the addition of a signal given from one time step to the next time step, namely the context (memory cell), represented by the symbol \( C_t \) in Figure 2.
number between zero and one; zero means that the information is totally blocked, while one means the entire information is included. The output from the sigmoid gate will be multiplied by another value to control how much the value is used.

For example, with the sigmoid gate in Figure 6, LSTM can manage how much information from $C_{t-1}$ is included into $C_t$.

D. Multinomial Logistic Regression (MLR)

Multinomial logistic regression is a logistic regression used when the dependent variable has a polichotomous or multinomial scale with nominal scale response variables with three categories [12]. For regression models with a dependent variable with a nominal category of three categories, the results of $Y$ variable are coded 1, 2, and 3. $Y$ variable is parameterized into three logit functions. Logistic regression method is stated in a probability model, namely the model in which the dependent variable is the logarithm of the probability that an attribute will apply in the condition of certain independent variables. As in [12], the model used in MLR is

$$\text{Logit}(P(Y = 1)) = \alpha + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

Using logistical transformation, logistic functions are obtained,

$$P_1(x) = \ln \left[ \frac{P(Y = 1) | x}{P(Y = 0) | x} \right] = \beta_{10} + \beta_{11} X_1 + \beta_{12} X_2 + \cdots + \beta_{1n} X_n$$

$$P_2(x) = \ln \left[ \frac{P(Y = 1) | x}{P(Y = 0) | x} \right] = \beta_{20} + \beta_{21} X_1 + \beta_{22} X_2 + \cdots + \beta_{2n} X_n$$

Based on the two logit functions, the trichotomous logistic regression model is obtained as follows:

$$\pi_0(x) = \frac{1}{1 + \exp(P_1(x)) + \exp(P_2(x))}$$

$$\pi_1(x) = \frac{\exp(P_1(x))}{1 + \exp(P_1(x)) + \exp(P_2(x))}$$

$$\pi_2(x) = \frac{\exp(P_2(x))}{1 + \exp(P_1(x)) + \exp(P_2(x))}$$

III. RESEARCH METHOD

The authors conducted research with the following stages:

A. Retrieval of Data

We perform tweet data retrieval through the service provided by Twitter, namely, Twitter API using the Python programming language by specifying key words or hashtags related to the topic taken until a number of tweets are needed.

B. Pre-Processing

We perform a cleaning process (Figure 7) on the tweet that has been obtained, including eliminating the URL, deleting the hashtag (#) and mention (), changing the negation word with the negation dictionary, deleting duplication of data and classifying the tweet using the Lexicon-based method, as seen in Figure 8 and Figure 9 with the opinion lexicon owned Hu and Liu [16], who divided tweets into positive, negative and neutral classes.
C. Processing

LSTM and MLR algorithms are applied. Tweets that have been cleaned and classified later are included in the classification method to be built. Classification methods to be built are LSTM and MLR. The accuracy results generated by the two classification methods are then calculated using the confusion-matrix method and then compared, so it can be determined which method can produce better accuracy values.

D. Visualization

The results of the information patterns found will be visualized and displayed in a form that is easier to understand, i.e. in the form of diagrams and tables.

E. Report Preparation

Writing and documenting the research starting from the initial stage, which is taking tweets to the results of sentiment analysis and visualizing sentiment analysis data into tables, diagrams, and wordcloud.

IV. IMPLEMENTATION AND RESULTS

Data obtained from Twitter by using access to the Twitter API that has undergone a pre-processing stage, namely cleaning and through the classification stage of the tweet using the Lexicon-based method, will be the datasets. It is processed at the processing stage. In this stage it is made classification methods by applying deep learning with LSTM algorithm and Machine-learning with MLR algorithm. In the final stage, the comparison of the two classification methods is visualized in the form of diagrams and tables.

A. Implementation of LSTM Classification Method

The built LSTM classification method is explained at this stage. This stage discusses briefly how to build a LSTM model from starting loading datasets to testing and visualization, as seen in Figure 10.

1) Designing a Layer on the LSTM Classification Method:

The built deep learning model has an input layer, hidden layer, and output layer. The input layer contains input data in the form of a matrix vector which is named the embedding-matrix variable, the hidden layer is an additional layer that is useful to train data repeatedly until it gets optimal accuracy or results, the output layer is the result of processing from the hidden layer. The following is the layer design on the LSTM classification method.

| TABLE I. LSTM DESIGN LAYER |
|-----------------------------|
| **Layer** | **Output Shape** | **Param #** |
| embedding_1 (Embedding) | embedding_1 (Embedding) | 4584400 |
| dropout_1 (Dropout) | (None, 35, 200) | 0 |
| lstm_1 (LSTM) | (None, 128) | 168448 |
| dense_1 (Dense) | (None, 64) | 8256 |
| dropout_2 (Dropout) | (None, 64) | 0 |
| activation_1 (Activation) | (None, 64) | 0 |
| dense_2 (Dense) | (None, 64) | 4160 |
| dropout_2 (Dropout) | (None, 64) | 0 |
| activation_2 (Activation) | (None, 64) | 0 |
| dense_3 (Dense) | (None, 3) | 195 |
| activation_3 (Activation) | (None, 3) | 0 |
Based on the design of the LSTM classification method layer in table I, the following layer design can be specified:

1) The input layer is an embedding matrix with a number of matrix vectors of 22,922 with a vector length of 35.
2) Hidden layer 1 has 64 neurons, uses the ReLU activation function, and the dropout is 0.4.
3) Hidden layer 2 has 64 neurons, uses the ReLU activation function, and the dropout is 0.4.
4) Output Layer has 3 neurons, using the Softmax activation function.

The Figure 11 shows a graph of the workflow model of the LSTM model built:

![Fig. 11. LSTM Model Workflow](image)

2) Training and Evaluation of the LSTM Classification Method: After declaration of input layer, hidden layer, and output layer, the next stage is the training data process. The training process is carried out as many as 3 epochs and the results of the training model will be stored every time there is an increase in value on the accuracy produced by each epoch. From the results of the training conducted with 3 epochs, it was found that the greatest accuracy value was produced at the 3rd epoch with an accuracy value of 0.9365 or 94%. Visualization of the results of accuracy and loss values during the training process can be seen in Figure 12.

![Fig. 12. Graph Comparison of Validation Accuracy against Train Accuracy](image)

3) Calculation of LSTM Accuracy with Confusion-Matrix: Accuracy calculations are obtained using the confusion-matrix method. After the data training is done through the learning process, and the test data is evaluated using validation data, the next step is to enter the LSTM model into a new variable to do prediction. Table II shows the results of accuracy calculation using the confusion-matrix method in the form of classification report.

|                  | Predicted_neg | Predicted_pos | Predicted_net |
|------------------|---------------|---------------|---------------|
| Negative         | 213           | 1             | 25            |
| Positive         | 0             | 192           | 14            |
| Neutral          | 8             | 8             | 312           |

|                  | Precision    | Recall       | F1-Score      | Support |
|------------------|--------------|--------------|---------------|---------|
| Negative         | 0.96         | 0.89         | 0.93          | 239     |
| Positive         | 0.96         | 0.93         | 0.94          | 206     |
| Neutral          | 0.89         | 0.95         | 0.92          | 328     |

Avg/Total 0.93 0.93 0.93 773

B. Implementation of MLR Classification Method

The built machine-learning classification method with MLR algorithm is explained at this stage. It discusses how to build a MLR model from starting loading datasets to testing and visualization, as seen in Figure 13. Following is the implementation chart of the MLR classification method:

1) MLR Classification Training Method: The training phase is carried out on several N-gram models to obtain the model with the best accuracy value which will then be evaluated with the Logistic Regression algorithm. N-gram is a method for retrieving bits of letter characters of n from a word. N-gram has three types of processing models in a sentence, the type of processing includes Unigram for separating one word in a sentence, Bigram for separating two words in a sentence, and Trigram for separating three words in a sentence. Classifier training models are carried out on the N-gram model with several conditions, among others, the Unigram with stop words model, Unigram without stop words, and Unigram without custom stop words. Custom stop words are stop words derived from the words that most often appear on the corpus. In table III it is shown the custom stopwords list in this study.

![Fig. 14. Comparison Chart of Results Using Stopwords, Without Stopwords and With No Custom Stopwords](image)
TABLE III. CUSTOM STOPWORDS LIST

|     | Negative | Positive | Neutral | Total |
|-----|----------|----------|---------|-------|
| the | 17705    | 10448    | 7185    | 35740 |
| to  | 10448    | 7522     | 4239    | 22099 |
| is  | 8138     | 5137     | 3133    | 16408 |
| you | 8216     | 4450     | 3228    | 15894 |
| and | 7655     | 4677     | 2671    | 15003 |
| of  | 7530     | 4609     | 2682    | 14921 |
| trump | 1673   | 10453    | 1629    | 13755 |
| in  | 5472     | 3322     | 1974    | 10768 |
| not | 4613     | 3298     | 2119    | 10030 |
| that| 4322     | 3155     | 1716    | 9193  |
| it  | 4069     | 2820     | 1718    | 8607  |
| this| 4260     | 2342     | 1765    | 8367  |
| are | 4434     | 2199     | 1674    | 8307  |
| for | 3424     | 2793     | 1352    | 7569  |
| he  | 3386     | 2523     | 1409    | 7318  |
| on  | 2818     | 1908     | 1104    | 5830  |
| with| 2565     | 1887     | 996     | 5448  |
| we  | 2301     | 1509     | 1289    | 5099  |
| do  | 2350     | 1499     | 1071    | 4920  |
| your| 2466     | 1201     | 865     | 4532  |

Based on the training process conducted using the Unigram word processing model with default stopwords, custom stopwords, and without stopwords, the best accuracy is obtained by 92% with default stopwords. After knowing the greatest accuracy using default stopwords, an experiment is conducted using the word processing Bigram and Trigram.

Figure 15 shows the results of accuracy comparison with Unigram, Bigram, and Trigram with stopwords training and validation processes, the best accuracy values for each N-Gram model are shown in the following table IV:

TABLE IV. ACCURACY VALUES FOR EACH N-GRAM MODEL

| Number of Feature | Validation Accuracy (%) |
|-------------------|-------------------------|
| Unigram          | 10000                   | 92                     |
| Bigram           | 10000                   | 91                     |
| Trigram          | 10000                   | 89                     |

From the training results the best accuracy of MLR classification method can be obtained through the Unigram word processing method with default stopwords with 92% accuracy.

2) Calculation of MLR Accuracy using Confusion-Matrix: Sentiment analysis carried out in this study is comparing the results of accuracy obtained from the method of deep learning classification using the LSTM algorithm and machine-learning classification method using MLR algorithm. Accuracy
calculations are obtained using the confusion-matrix method. The table V shows the results of accuracy calculations with Unigram and default stopwords using the confusion-matrix method with 10,000 features in the form of classification reports:

|                  | Predicted_neg | Predicted_pos | Predicted_net |
|------------------|---------------|---------------|---------------|
| Negative         | 213           | 0             | 16            |
| Positive         | 0             | 175           | 15            |
| Netral           | 15            | 18            | 320           |

|       | Precision | Recall | F1-Score | Support |
|-------|-----------|--------|----------|---------|
| Negative | 0.93     | 0.93   | 0.93     | 229     |
| Positive  | 0.91     | 0.92   | 0.91     | 190     |
| Netral    | 0.91     | 0.91   | 0.91     | 353     |
| Avg/Total | 0.92     | 0.92   | 0.92     | 772     |

C. Comparison of Accuracy Results for the LSTM and MLR Classification Methods

The final result in this study is to determine which classification method is better in conducting sentiment analysis. Based on the classification report obtained it can be concluded that the Deep Learning classification method with the LSTM algorithm is better in analyzing sentiments compared to the MLR classification method using Unigram with default stopwords. The following table VI shows the comparison results of the two classification methods tested.

|                  | Precision | Recall | F1-Score | Support |
|------------------|-----------|--------|----------|---------|
| Negative         | MLR 0.93  | LSTM 0.96 | 0.93 | 0.93 | 0.93 |
| Positive         | MLR 0.91 | LSTM 0.96 | 0.92 | 0.93 | 0.92 | 0.94 |
| Netral           | MLR 0.91 | LSTM 0.89 | 0.91 | 0.95 | 0.91 | 0.92 |
| Total            | 0.92     | 0.93   | 0.93     | 0.92   | 0.93 |

D. Testing of LSTM and MLR Classification Methods

This test is carried out by providing input data on the two classification methods that have been built and have completed the learning process with training data. For example, the author will enter three sentences that later the LSTM and MLR classification method will provide the results of classification of sentiments. The following table VII shows three sentences that will be used as testing material:

TABLE VII. EXAMPLES OF SENTENCE TO TEST THE CLASSIFICATION METHOD

| Sentence     | Classification |
|--------------|----------------|
| Bob Mueller gives him the information he needs to hold Putin accountable for | Netral(2) |
| Trump won the American presidential election | Positive(1) |
| 12 Russian programmers have hacked the US presidential election system | Neutral(0) |

The sentiment analysis results are stored in the variable `LSTMpredic` in the array for the LSTM classification method and the `MLRpredic` variable for the MLR classification method. Sentiment analysis results are stored in the ‘LSTMpredic’ variable. The following table VIII shows the result of sentiment classification using the LSTM classification method.

|       | 0     | 1     | 2     |
|-------|-------|-------|-------|
| 0     | 0.925665 | 0.000798 | 0.073537 |
| 1     | 0.000000 | 0.999739 | 0.000260 |
| 2     | 0.004075 | 0.002234 | 0.993691 |

Table IX shows the results of the sentiment classification obtained using the MLR classification method. The best accuracy results obtained from the MLR classification method are using the Unigram word processing model with default stopwords and 10,000 features. The test was carried out by entering the same three sentences as the tests performed on the LSTM classification model. This was done so that the accuracy of the resulting sentiment classification could be compared.

E. Comparison Results of Sentiments on Trial Data

Comparison of classification results can be seen in table X as follows:

|       | Negative(0) | Positive(1) | Netral(2) | Cl. |
|-------|-------------|-------------|-----------|-----|
| Sent. | LSTM        | MLR         | LSTM      | MLR | Cl. |
| 1     | 0.926       | 0.707       | 0.001     | 0.026 | 0.074 | 0.267 | 0 |
| 2     | 0.000       | 0.000       | 0.999     | 0.924 | 0.0002 | 0.976 | 1 |
| 3     | 0.004       | 0.018       | 0.002     | 0.027 | 0.994 | 0.954 | 2 |

The conclusions obtained based on the table on the comparison of LSTM and MLR classification results are as follows:

1) The sentence “12 Russian programmers have hacked the US presidential election system" contains negative sentiment values with a weight of 0.926 (92%) for the LSTM classification method and a weight of 0.707 (70%) for the MLR classification method.

2) The sentence “Trump won the American presidential election" contains a positive sentiment value with a weight of 0.999 (99%) for the LSTM classification method and a weight of 0.924 (92%) for the MLR classification method.

3) The sentence “Bob Mueller has given him the information he needs to hold Putin accountable for” contains a neutral sentiment value with a weight of 0.994 (99%) for the LSTM classification method and a weight of 0.954 (95%) for the MLR classification model.
4) Based on the sentiment class, the LSTM classification method and MLR are able to produce the same sentiment class, but the weighting of sentiment value from the LSTM classification model is better.

5) The LSTM classification method is better than the MLR classification method in classifying sentiment classes.

V. Conclusion and Future Work

In this study, sentiment analysis was carried out on the topic "Russian Hacking Cases Regarding the 2016 US Presidential Election". Sentiment analysis was carried out by testing the classification method of deep learning using LSTM algorithm and Machine-learning using MLR algorithm. Based on the results obtained from testing the LSTM- and MLR-classification method, the following conclusions can be drawn: Testing of the Deep-learning classification method using the LSTM algorithm produces an accuracy value of 93%. Testing of the Machine-learning classification method using the MLR algorithm produces an accuracy value of 92%. Deep-learning classification method using LSTM algorithm is better in doing sentiment analysis than Machine-learning classification method using MLR algorithm. Differences in accuracy generated by the classification method of LSTM Deep-learning and MLR Machine-learning are significant enough.

The sentiment analysis carried out in this study still has many shortcomings. For that it is necessary to develop the improvement of the application that has been made in this study. The suggestions of application development are as follows:

1) The research was conducted using only two classification methods, namely LSTM deep learning algorithm and MLR machine-learning algorithm. It is expected that the other classification method will be added so that more classification methods can be compared in the it’s accuracy results.

2) The dataset used is still relatively small for data processing with deep learning algorithms. For further development, the dataset used is expected to be even more so that the resulting accuracy can be better.

Acknowledgment

The Authors gratefully acknowledge Gunadarma University for providing research funding and for permission in using the research facilities.

References

[1] B. Agarwal and N. Mittal, Prominent Feature Extraction for Sentiment Analysis. Springer, 2016.
[2] J. Bayer and C. Osendorf, “Learning stochastic recurrent networks,” in NIPS 2014 Workshop on Advances in Variational Inference, 2014.
[3] E. Cambria, D. Das, S. Bandyopadhyay, and A. Feraco, Eds., A Practical Guide to Sentiment Analysis. Springer, 2017.
[4] A. Chaudhuri, Visual and Text Sentiment Analysis through Hierarchical Deep Learning Networks. Springer, 2019.
[5] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” Dec. 2014. [Online]. Available: https://arxiv.org/pdf/1412.3555.pdf
[6] J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, and Y. Bengioy, “A recurrent latent variable model for sequential data,” Jun. 2015. [Online]. Available: arxiv.org/pdf/1506.02216v3.pdf
[7] M. Dehaff, “Sentiment analysis, hard but worth it!” on-line, 2010. [Online]. Available: http://www.customerthink.com/blog/sentiment_analysis_hard_but_worth_it/
[8] F. Gers and E. Schmidhuber, “Lstm recurrent networks learn simple context-free and context-sensitive languages,” IEEE Transactions on Neural Networks, vol. 12, no. 6, pp. 1333 – 1340, Nov. 2001.
[9] F. Gers and J. Schmidhuber, “Recurrent nets that time and count,” in Proceedings of the IEEE-ENNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000), Neural Computing: New Challenges and Perspectives for the New Millennium. IEEE, Jul. 2000.
[10] A. Hepburn, “Infographic: Twitter statistics, facts & figures,” on-line, 2010. [Online]. Available: http://www.digitalbuzzblog.com/infographic-twitter-statistics-facts-figures/.
[11] J. Hill, “Sentiment analysis using graphlab,” vol. 8, no. 6, pp. 1155–1158, Jun. 2017.
[12] M. M. Nasr, E. M. Shaaban, and A. M. Hafez, “Building sentiment analysis model using graphlab,” International Journal of Scientific & Engineering Research, vol. 8, no. 6, pp. 1155–1158, Jun. 2017.
[13] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[14] O. Kolchyna, T. T. P. Souza, P. C. Treleaven, and T. Aste, “Twitter sentiment analysis: Lexicon method, machine learning method and their combination,” On-Line, Sep. 2015. [Online]. Available: https://arxiv.org/abs/1507.00955
[15] B. Liu, In Handbook of natural language processing, 2nd ed., 2010, vol. 2, ch. Sentiment Analysis and Subjectivity, pp. 627–666.
[16] B. Liu, Sentiment Analysis: Mining Opinions, Sentiments, and Emotions. Cambridge University Press, 2015.
[17] M. M. Nasr, E. M. Shaaban, and A. M. Hafez, “Building sentiment analysis model using graphlab,” International Journal of Scientific & Engineering Research, vol. 8, no. 6, pp. 1155–1158, Jun. 2017.
[18] C. Olah, “Understanding lstm networks,” on-line, Aug. 2015. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
[19] S. Poria, A. Hussain, and E. Cambria, Multimodal Sentiment Analysis. Springer, 2018.
[20] L. Vu and T. Le, “A lexicon-based method for sentiment analysis using social network data,” in Int’l Conf. Information and Knowledge Engineering, 2017.
[21] D. Zarella, The Social Media Marketing Book. PT Serambi Ilmu Semesta Anggota IKAPI, Jakarta, 2010.