Sentiment Analysis on Hotel Ratings Using Dynamic Convolution Neural Network

Novendra Adisaputra Sinaga[1], Teddy Surya Gunawan [2], Wanayumini [3]

[1,2,3]*Magister of Computer Science, Potensi Utama University*
JL. KL. Yos Sudarso Km. 6,5 No. 3-A, Medan
[1]mrnoven@gmail.com
[2]tsgunawan@gmail.com
[3]wanayumini@gmail.com

***Abstract*— Currently, the role of information technology is very important in everyday life because heavy workloads can become easier, communication time can be made shorter and data processing can be faster and more accurate. Hotel ranking sentiment analysis can provide important information for hotel owners and managers to improve the quality of service and guest experience. It can also be used by prospective guests to make the right booking decisions. Sentiment analysis can identify positive or negative feelings from guest reviews. There are 694,213 data reviews about hotels using English which are used as training data. The data was preprocessed and 76,905 vocabularies were obtained by utilizing Word2Vec. The training data was carried out at the encoding stage. The DCNN model is given a K-Max-Polling value of 2. The model is trained for 20 epochs. The model that has been formed is tested with 173,554 data and obtained an accuracy rate of 95%.**

***Keywords*— sentiment analysis, word2vec, dynamic convolution neural network**

I. INTRODUCTION

Currently, the role of information technology is very important in everyday life because heavy workloads can become easier, communication time can be made shorter and data processing can be faster and more accurate [1][2]. With the development of information technology, the development of online applications is currently very rapid. There are many applications available for various needs, ranging from entertainment to productivity. One feature that is commonly found in online applications is the rating and comments of the product. This feature provides an opportunity for users to share their opinions and feelings about the application. The rating can provide insight into how well the application is received by users. Users can give star ratings, which indicate how well they rate the application. Star ratings can provide a clear view of how well the application is received by users [3]. Comments, on the other hand, provide an opportunity for users to give more detailed opinions about the application. Users can express their feelings and opinions about the features of the application, ease of use, or problems they may have encountered. Comments can also give an insight into how the application compares to similar applications. Analyzing public information from social networking sites can yield interesting results and insights about public opinion on almost any product, service, or behavior[4][5], [6].

Hotel ranking sentiment analysis can provide important information for hotel owners and managers to improve the quality of service and guest experience. It can also be used by prospective guests to make the right booking decisions. Sentiment analysis can identify positive or negative feelings from guest reviews. This can provide a clear view of what is well received by guests and what needs improvement. For example, if guests complain about poor service, hotel managers can improve staff training or increase the number of staff to enhance customer service. In addition, sentiment analysis can also be used to uncover trends in guest reviews. This can provide insights into how the hotel's performance changes over time and provide a view of how competitor hotels compare to the hotel. Sentiment analysis can also be used to enhance hotel marketing. For example, if guests complain about high prices, hotel managers can offer promotions or discounts to attract more guests. Overall, hotel ranking sentiment analysis can provide important insights for hotel owners and managers to improve the quality of service and guest experience, and help prospective guests make the right booking decisions.

Deep Learning is a rapidly developing field of Machine Learning that processes data based on layers or several stages of nonlinear information processing in a hierarchical manner. Ensemble Learning and Deep Learning can yield promising results for sentiment analysis tasks [7][8].The Convolutional Neural Network (CNN) approach has static convolutional kernels (not adaptive to input) but dynamic network architecture (increasing parameters, number of layers, number of channels, etc.). Dynamic Convolutional Neural Network (DCNN) presents dynamic convolutional kernels (adaptive to input [conv = f(x)]) with static network architecture (constant depth and width). The striking difference between CNN and DCNN is that DCNN receives two inputs. The first input is the feature map from the previous layer and the second is the filter. The feature map is obtained from the input by following sub-network A. The filter is the result of applying a separate convolutional sub-network B on the input. The output of the layer is calculated by combining the filter over the feature map of the previous layer in the same way as in the convolutional layer. The function of the filter here is the function of the input and therefore varies from one sample to another [9]. DCNN is a new operator design that improves model complexity without changing the depth or width of the network by combining multiple kernels dynamically based on attention dependent on the input [10]. This method achieved significant accuracy improvement with fewer kernels per layer, smaller model sizes, and additional computational costs that can be ignored [11][12].

In the research [13] entitled Sentiment Analysis of Movie Opinion in Twitter Using Dynamic Convolutional Neural Network Algorithm, it was found that the Dynamic Convolutional Neural Network algorithm had better accuracy compared to the Naive Bayes method, with an accuracy value of 80.99%, while the accuracy value generated by Naive Bayes was 76.21%.

In this study, an effort to improve accuracy will be made by adding to the transformation stage using the word2vec method to obtain feature extraction from data.

II. METHODS

A. World Embedding

Word2Vec is a Neural Networks Language Model (NNLM) update that can preserve the linear order of words. The Continous Bag-of-Words (CBOW) and Continous Skip-gram algorithms are utilized in Word2Vec. Word2Vec has the benefit over other word embedding models in that it trains more quickly, is more effective, and can handle big datasets. Another benefit is that Word2Vec can solve the word imbalance issue since it is efficient at capturing semantic links between words, whereas other approaches fall short [14][15][16], [17].

B. Dynamic Convolution Neural Network (DCNN)

Dynamic Convolutional Neural Network (DCNN) is an algorithm that uses a convolutional architecture that alternates between wide convolutional layers and dynamic pooling layers determined by dynamic k-max pooling. Because the width of the middle layer feature maps varies according to the length of the input sentence, the resulting architecture is called Dynamic Convolutional Neural Network [13].

1) *Wide*: Used to obtain the first matrix of DCNN, it requires word embeddings $w_i \in R^d$ for each word in the sentence and the input sentence matrix $s \in R^{d \times s}$. The value of the word embedding is a parameter that is optimized during training. The convolutional layer is obtained from the convolution process between the weight matrix $m \in R^{d \times m}$ and the activation matrix on the previous layer. Thus, a wide convolution matrix with dimension d x (s + m -1) is obtained.
2) *K-Max Pooling*: Applied after the last convolutional layer. K-max pooling is used to reduce the length of different sample vectors with the same length as the fully connected layer. Therefore, k-max pooling is applied after the top convolutional layer. K-max pooling is taken from the k-max in sequence, not a single maximum value.

K-Max Pooling is applied after the last convolutional layer. K-max pooling is used to reduce the length of different sample vectors with the same length as the fully connected layer. Therefore, k-max pooling is applied after the top convolutional layer. K-max pooling is taken from the k-max in sequence, not a single maximum value.

Dynamic K-Max Pooling is a k-max pooling operation where k is a function of the sentence length and network depth. The pooling parameter can be seen in equation 1 below:

$$k_l = max(k_{top} \left\lceil \frac{L-l}{L} S \right\rceil) \quad \cdots\cdots\cdots\cdots 1$$

Where l is the number of convolutional layers currently in progress when pooling is applied. L is the total number of convolutional layers in the network, k_top is a fixed pooling parameter for the topmost convolutional layer, and S is the sentence length. The DCNN for a seven-word input sentence can be seen in Fig. 1 below:

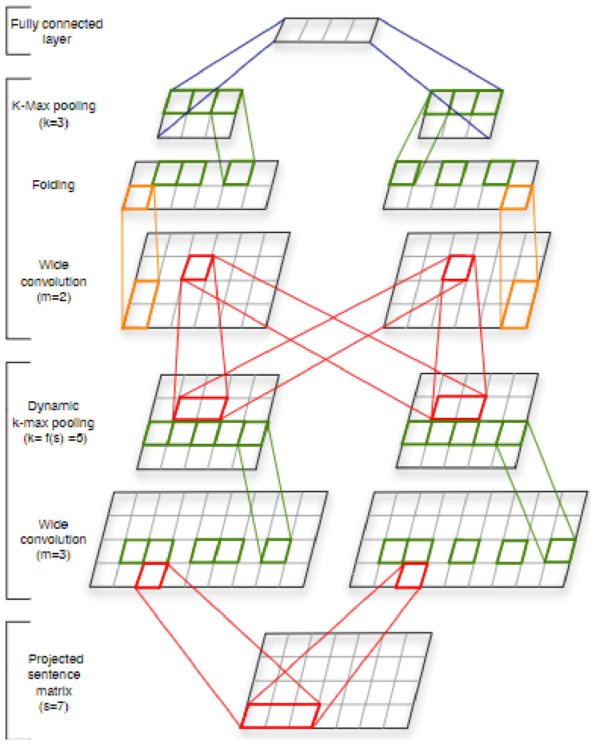

Fig.1. DCNN for seven word inputs of Sentences

From Fig. 1, the word embedding has a size of 4. The network has two convolutional layers with two feature maps each. The width of the filter on the two layers is 3 and 2 respectively. The k-max pooling layer (dynamic) has k values of 5 and 3.

3) *Folding*: Applied after the last convolutional layer and before the k-max pooling layer by summing every two lines on the feature map. For the line map,

the line map is folded back by $(d/2)$, reducing the representation size by half.

III. METHODOLOGY

The Research Methodology steps can be seen in Fig. 2. below:

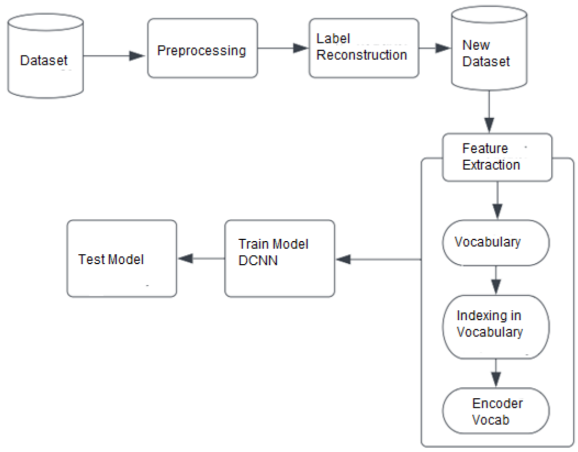

Fig.2. The Research Methodology Step

The stages of research methodology can be seen in Fig. 2 above. Training and testing data are carried out in the preprocessing stage. After the preprocessing stage is complete, the next stage is the feature extraction stage by utilizing the word2vec method. The training data will be computed using the DCNN algorithm which will produce a sentiment model. In the data testing stage, the word2vec value after going through the previous stages will enter the sentiment model stage which will be the predicted value of the sentiment test from the assessment. In the next stage, the accuracy results of the test will be obtained. The stages that are carried out are:

A. Data Source

Implementing the DCNN algorithm requires data to be trained on the data. The data used is secondary data obtained from Kaggle. The dataset consists of 17 columns and 867,767 records. The fields of the Dataset can be seen in Fig.3 below:

```
Hotel_Address                                  object
Additional_Number_of_Scoring                    int64
Review_Date                                    object
Average_Score                                 float64
Hotel_Name                                     object
Reviewer_Nationality                           object
Negative_Review                                object
Review_Total_Negative_Word_Counts               int64
Total_Number_of_Reviews                         int64
Positive_Review                                object
Review_Total_Positive_Word_Counts               int64
Total_Number_of_Reviews_Reviewer_Has_Given      int64
Reviewer_Score                                float64
Tags                                           object
days_since_review                              object
lat                                           float64
lng                                           float64
dtype: object
```

Fig. 3. Field Data Source

B. Preprocessing

Before processing the data, the data cleaning stage is carried out. Cleaning Data aims to convert into lowercase letters, eliminate unnecessary characters such as [.], [,], [!] other characters also replace words such as examples: "ain't" becomes "am not", "aren't" becomes "are not", "can't" becomes "cannot".

C. New Dataset

After Proprocessing, the next step is to combine 2 fields into 1 and set the labeling process as shown in Fig.4. below:

	reviews	labels
0	cold room light switch knob missing warm towe...	0
1	i don t really have any complaints	0
2	i loved the room and the appearance of the ho...	1
3	we liked everything the staff were very atten...	1
4	the size of the bathroom was abit cramped	0
...	...	...
867762	great location and staff very helpful	1
867763	nothing	0
867764	staff	1
867765	very busy hotel no personal touch	0
867766	location and facilities	1

867767 rows × 2 columns

Fig.4. New Dataset

D. Vocabulary

After structuring the dataset, field reviews are carried out in word mining, namely vocabulary by utilizing the word2vec model. The vocabulary formed from the data is as follows: *'evangelo', 'helped', 'us', 'to', 'exchange', 'our', 'e', 'small', 'money', 'more', 'than', 'my', 'expectancy', 'get', 'such', 'a', 'service', 'from', 'the', 'hotel', 'staff', 'no', 'need', 'worry', 'reach', 'everywhere', 'in', 'city', 'or', 'going', 'airport', 'min', 'train', 'station', 'supermarket', 'everything', 'is', 'easy', 'very', 'close', 'stadium'.*

E. Indexing in Vocabulary

Each word in the vocabulary can be assigned an index to facilitate data processing. An example of indexing can be seen in Fig. 5 below:

```
vocabulary = ['evangelo', 'helped', 'us', 'to', 'exchange', 'our', 'e', 'small', 'money', 'more',
'than', 'my', 'expectancy', 'get', 'such', 'a', 'service', 'from', 'the', 'hotel', 'staff', 'no',
'need', 'worry', 'reach', 'everywhere', 'in', 'city', 'or', 'going', 'airport', 'min', 'train',
'station', 'supermarket', 'everything', 'is', 'easy', 'very', 'close', 'stadium']

word_to_index = {word: index for index, word in enumerate(vocabulary)}
index_to_word = {index: word for index, word in enumerate(vocabulary)}
```

Fig. 5. Indexing Vocabulary

The source code above displays the index on the vocabulary as follows: 'evangelo': 0, 'helped': 1, 'us': 2, 'to': 3, 'exchange': 4, 'our': 5, 'e': 6, 'small': 7, etc)

F. Encoder Vocabulary

Vocabulary Encoder is a method to convert text into a numerical representation that can be processed by machines. It aims to help machines understand the text better. By converting text into a numerical representation, machines can understand and process text better than by only processing text as a string. The Encoder stage can be seen in the snippet in Fig. 6 below:

```
teks = "exchange"
oneHotKata = tokenizer.texts_to_sequences(teks)
encodedKata =pad_sequences(oneHotReviews, maxlen=40, padding='post')

print(encodedKata)
```

Fig.6. Encoder Vocabulary

The source code above will produce the following numpy array:

```
[[ 1118 20543     7 ...     0     0     0]
 [   12     3    87 ...     0     0     0]
 [   67  2485    53 ...     0     0     0]
 ...
 [    8    27     2 ...     0     0     0]
 [  281    11   494 ...     0     0     0]
 [ 6644    55   194 ...     0     0     0]]
```

G. DCNN Model

The next step is to model the DCNN architecture with 1D convolution utilizing the hard framework and by utilizing the global_k_max_polling1D function that has been customized as in Table 1 below:

TABLE I
ARSITEKTUR MODEL DCNN

Layer (Type)	Output (Shape)	Param
Embedding	(None, 40, 128)	9843840
Conv1D	(None, 40, 256)	65792
Global_K_Max_Polling	(None, 768)	0
Dence	(None, 128)	98432
Dence	(None, 1)	129

The Explanation of DCNN Architecture in Table 1 as follows:

1) Embedding Layer: This is the input layer to map each token in the vocabulary to the vector representation. The output has the form (None, 40, 128), meaning the number of rows can't be specified (marked with "None"), 40 is the length of the input text and 128 is the size of the vector representation. The parameters in this layer are 9,843,840.
2) *Conv1D Layer*: This is a 1D convolution layer, which works to identify features in the input data. The output has the form (None, 40, 256), meaning each row has 40 features, and each feature has a size of 256. The parameters in this layer are 65,792
3) *Global_K_Max_Polling layer:* This is the pooling layer, which serves to reduce the dimensionality of the data, focusing on the most important features. The output has the form (None, 768), meaning each row has 768 features. There are no parameters in this layer.
4) *Dence Layer:* This is a fully connected layer, which serves to make connections between all neurons in the previous layer. The output has the form (None, 128), meaning that each row has 128 neurons. The parameter in this layer is 98,432.
5) *Dence layer:* This is an additional fully connected (dense) layer, which serves to predict the model output. The output has the form (None, 1), meaning that each row has 1 prediction. The parameters in this layer are 129.

H. Accuracy Result

The final step after building the model is to check the accuracy of the formed model. The accuracy values that are checked are Accuracy, Precision, Recall, and F-Measure.

$$Accuracy = \frac{TP+TN}{Predicted+Actual} = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

$$Precision = \frac{TP}{Positive\ Predicted} = \frac{TP}{TP+FP} \quad (3)$$

$$Recall = \frac{TP}{Negative\ Predicted} = \frac{TP}{TP+FN} \quad (4)$$

III. RESULT

A. *Model Performance every Epoch*

From the results of training and evaluation of the model, the results obtained are shown in Table II below:

TABLE II
ACCURACY AND LOSS VALUE

Epoch	Loss	Accuracy	val_loss	val_accuracy
1	0.1980	0.9398	0.1668	0.9392
2	0.1443	0.9507	0.1568	0.9458
3	0.1352	0.9545	0.1511	0.9494
4	0.1295	0.9568	0.1541	0.9464
5	0.1269	0.9579	0.1714	0.9472
6	0.1255	0.9584	0.1565	0.9485
7	0.1249	0.9589	0.1578	0.9451
8	0.1244	0.9598	0.1619	0.9451
9	0.1242	0.9600	0.1562	0.9479

10	0.1249	0.9596	0.1655	0.9452
11	0.1251	0.9600	0.1630	0.9467
12	0.1256	0.9599	0.1647	0.9461
13	0.1254	0.9600	0.1612	0.9452
14	0.1279	0.9590	0.1629	0.9461
15	0.1267	0.9600	0.1730	0.9459
16	0.1283	0.9595	0.1703	0.9414
17	0.1284	0.9597	0.1701	0.9456
18	0.1291	0.9587	0.1733	0.9381
19	0.1297	0.9585	0.1654	0.9469
20	0.1305	0.9593	0.1629	0.9464

In Table I above, 20 Epochs were performed to calculate the Accuracy and Loss values for both the training data and validation data, and the results show that as the number of epochs increases, the accuracy level also increases, and as the number of epochs increases, the loss value decreases.

B. *Confusion Matrix*

The model that has been obtained is then tested on test data. The testing process in this study uses performance evaluation, namely the Confusion Matrix method in Fig. 7 below:

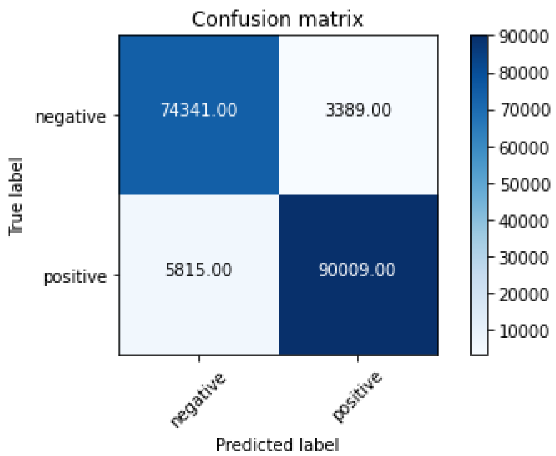

Fig.7. Confusion Matrix

Fig. 7 explains the confusion matrix. The confusion matrix is a tabular representation of the model predictions and the actual value of the data. In this case, the confusion matrix displays the number of data predicted as Negative and the number of data predicted as Positive. In the above case, the actual number of Negative data is 74,341 and the model predicts 3,389 data as Positive. Meanwhile, the actual number of Positive data is 90,009 and the model predicts 5,815 data as Negative. From The confusion matrix above, it is obtained:

TP = 90.009
TN = 5.815
FP = 74.341
FN = 3.389

C. *Accuracy Results*

In Table 3 are the evaluation results of the DCNN Model using Word2Vec at the preprocessing stage below:

TABLE III
THE EVALUATION RESULTS

Label	**Precision**	**Recall**	**F1 Score**	**Support**
0	0.93	0.96	0.94	77.730
1	0.96	0.94	0.95	95.824
Accuracy			0.95	173.554
macro av	0.95	0.95	0.95	173.554
weighted avg	0.95	0.95	0.95	173.554

In Table III above, it can be concluded that the precision, recall, and f1-score values for the negative sentiment (0) are respectively 0.93, 0.96, and 0.94. Whereas, the precision, recall, and f1-score values for the positive sentiment (1) are respectively 0.96, 0.94, and 0.95. And the Accuracy rate of the DCNN model is 95%.

D. *Accuracy and Validation Accuracy Value Chart*

This graph visualizes the change in accuracy value and validation accuracy value during the training process, helping to understand how the model evolves and improves its performance. The accuracy value indicates how well the model predicts data labels on the training dataset, while the validation accuracy value indicates how well the model predicts data labels on the validation dataset. Comparison of Accuracy and Validation Accuracy values can be seen in Fig. 8 below:

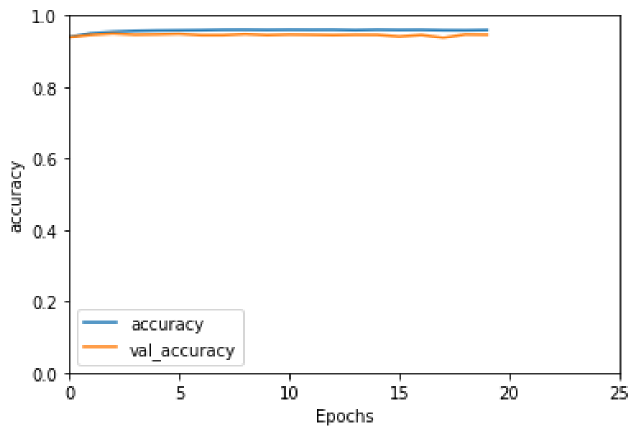

Fig. 8. Comparison of Accuracy and Validation Accuracy

In Fig. 8 above, it can be seen that the accuracy value is better than the accuracy validation value with a difference of 1%.

E. Graph of Loss Value and Loss Validation

Loss and Validation Loss value graphs are visualizations that show the change in loss (or loss) values on training data and validation data during the model training process. Loss value is a metric used to measure how far the model predicts from the target. The smaller the loss value, the better the model's prediction of the target. A Comparison of Loss Value and Validation Loss can be seen in Fig. 9 below:

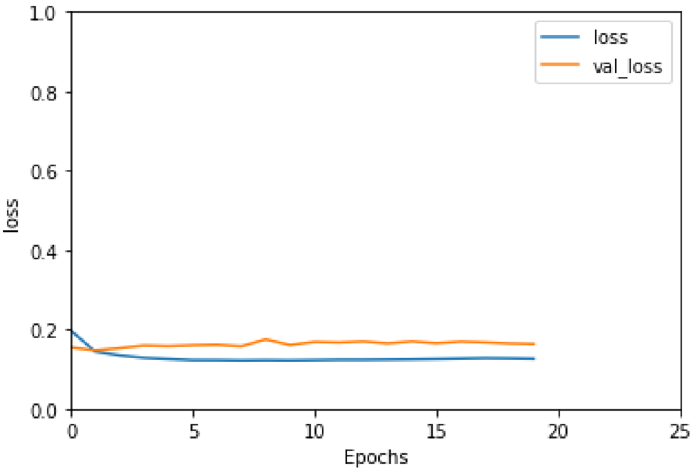

Fig. 9. Loss value and Val_Loss value

In Fig. 9 above, it can be seen that the accuracy value is better than the accuracy validation value with a difference of 3%.

V. RESULTS

There are 694,213 data reviews about hotels using English which are used as training data. The data is carried out in a preprocessing stage and obtained 76,905 vocabularies by utilizing Word2Vec. The training data is carried out in the encoding stage. The DCNN model is given a K-Max-Polling value of 2. The model is trained for 20 epochs to get a model with a high accuracy value. There are 2 sentiments tested in this study, namely positive sentiment and negative sentiment represented by values 1 and 0. The model that has been formed is tested with 173,554 data and obtained an accuracy rate of 95%. This model can be further developed by utilizing other preprocessing models with different data and languages to test whether the accuracy of the model remains stable and also develop the model so that the accuracy validation value is higher than the accuracy and the loss validation value is lower than the loss value.

REFERENCES

[1] R. Roslina, S. Saulan, and Wanayumini, "Model of Information Technology Facility Service Based on user Satisfaction," *J. Phys. Conf. Ser.*, vol. 1361, no. 1, 2019, doi: 10.1088/1742-6596/1361/1/012032.

[2] N. A. Sinaga *et al.*, "Decision support system with MOORA method in selection of the best teachers," *AIP Conf. Proc.*, vol. 2453, 2022, doi: 10.1063/5.0094437.

[3] A. P. Thenata, "Text Mining Literature Review on Indonesian Social Media," *J. Edukasi dan Penelit. Inform.*, vol. 7, no. 2, p. 226, 2021, doi: 10.26418/jp.v7i2.47975.

[4] T. S. Gunawan *et al.*, "Development of video-based emotion recognition using deep learning with Google Colab," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 18, no. 5, pp. 2463–2471, 2020, doi: 10.12928/TELKOMNIKA.v18i5.16717.

[5] A. H. Alamoodi *et al.*, "Sentiment analysis and its applications in fighting COVID-19 and infectious diseases: A systematic review," *Expert Syst. Appl.*, vol. 167, p. 114155, 2021, doi: 10.1016/j.eswa.2020.114155.

[6] I. E. Firdausi, I. Mukhlash, A. D. S. Gama, and N. Hidayat, "Sentiment analysis of customer response of telecommunication operator in Twitter using DCNN-SVM Algorithm," *J. Phys. Conf. Ser.*, vol. 1490, no. 1, 2020, doi: 10.1088/1742-6596/1490/1/012071.

[7] A. Onan, "Sentiment analysis on massive open online course evaluations: A text mining and deep learning approach," *Comput. Appl. Eng. Educ.*, vol. 29, no. 3, pp. 572–589, 2021, doi: 10.1002/cae.22253.

[8] M. M. Islam, J.-H. Baek, and N. Tasnim, "Gender Prediction from Masked Face Images using Deep Learning towards Smart Store Customer Management," no. February, pp. 67–71, 2021, [Online]. Available: https://www.researchgate.net/publication/350106387.

[9] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A Convolution Neural Network for Modelling Sentences." 2014.

[10] A. Bhujel and D. R. Pant, "Dynamic Convolutional Neural Network For Image Super-resolution," *J. Adv. Coll. Eng. Manag.*, vol. 3, pp. 1–10, 2018, doi: 10.3126/jacem.v3i0.18808.

[11] Y. Chen, X. Dai, M. Liu, D. Chen, L. Yuan, and Z. Liu, "Dynamic convolution: Attention over convolution kernels," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 11027–11036, 2020, doi: 10.1109/CVPR42600.2020.01104.

[12] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *52nd Annu. Meet. Assoc. Comput. Linguist. ACL 2014 - Proc. Conf.*, vol. 1, pp. 655–665, 2014, doi: 10.3115/v1/p14-1062.

[13] F. Ratnawati and E. Winarko, "Sentiment Analysis of Movie Opinion in Twitter Using Dynamic Convolutional Neural Network Algorithm," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 12, no. 1, p. 1, 2018, doi: 10.22146/ijccs.19237.

[14] H. Juwiantho *et al.*, "Sentiment Analysis Twitter Bahasa Indonesia Berbasis WORD2VEC Menggunakan Deep Convolutional Neural Network," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 7, no. 1, pp. 181–188, 2020, doi: 10.25126/jtiik.202071758.

[15] Q. Shen, Z. Wang, and Y. Sun, "Sentiment analysis of movie reviews based on CNN-BLSTM," *IFIP Adv. Inf. Commun. Technol.*, vol. 510, pp. 164–171, 2017, doi: 10.1007/978-3-319-68121-4_17.

[16] Y. K. Saheed, M. A. Hambali, M. O. Arowolo, and Y. A. Olasupo, "Application of GA Feature Selection on Naive Bayes, Random Forest and SVM for Credit Card Fraud Detection," *2020 Int. Conf. Decis. Aid Sci. Appl. DASA 2020*, pp. 1091–1097, 2020, doi: 10.1109/DASA51403.2020.9317228.

[17] W. Widayat, "Analisis Sentimen Movie Review menggunakan Word2Vec dan metode LSTM Deep Learning," *J. Media Inform. Budidarma*, vol. 5, no. 3, p. 1018, 2021, doi: 10.30865/mib.v5i3.3111.