HYBRID CNN AND RNNS MODEL FOR SENTIMENT ANALYSIS

Andreas Permana Putra Sitanggang
Program Studi Teknik Informatika Fakultas Ilmu Komputer, Universitas Katolik Soegijapranata
19k10063@student.unika.ac.id

ABSTRACT

In the realm of sentiment analysis, understanding public opinion and customer feedback is of paramount importance. This study researches into the performance of a hybrid model that fuses Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for sentiment analysis, considering various datasets. The findings consistently highlight the remarkable enhancement in sentiment classification accuracy achieved by the hybrid model in comparison to basic models, owing to its ability to capture complex data patterns. While the hybrid models tend to require slightly more training time, the trade-off between accuracy and training time remains manageable. Furthermore, the hybrid models outperform basic models across datasets.

Keywords: sentiment analysis, LSTM, GRU, CNN, accuracy

1. INTRODUCTION

A lot of people use Facebook, Instagram, Twitter, and other social media to express their thoughts and feelings. Those make massive amounts of data on various topics to be generated every day. With this data, many organizations (e.g., government, academia) conduct text mining or NLP (Natural Language Processing) to gain valuable information such as news analysis, prediction election results, market research, customer feedback analysis, etc. One popular of research in NLP is sentiment analysis. sentiment analysis, also known as opinion mining is a computational approach that uses identifying, extracting, and classifying sentiments expressed in text data.

Many studies on sentiment analysis have been conducted in recent years, The approach or method that is often used is traditional machine learning methods such as the Lexicon-based method [5], Naive Bayes, Maximum Entropy, and Support Vector Machine (SVM) [2]. In addition to the machine learning approaches listed, deep learning methods such as BERT, Convolution Neural Network (CNN), and Recurrent Neural Network (RNN), that each has promising results in NLP tasks [3], [4], [6], [7]. In the last few years, deep learning become more popular for several reasons. One of them is CNN the ability to capture local and global dependencies in the input data and LSTM and GRU have performed well in long-range dependencies in sequential data. Another advantage is when working with a large dataset [9]. Machine learning methods have the disadvantages of high training time and low accuracy results while LSTM can deal with the vanishing gradient problem that is common in long data processing.

Recent research has explored hybrid models that combine CNN and LSTM deep learning architectures for sentiment analysis [10], [11]. These hybrid models aim to leverage the strengths of both approaches to enhance overall performance. In this study, we implemented sentiment analysis with a hybrid deep learning method by combining RNNs and the CNN model on a labeled dataset containing positive and negative sentiments sourced from Kaggle. We focused on evaluating the performance of the hybrid model by comparing its test accuracy and time training against the individual model.

2. METHODOLOGY

2.1. Data Collections

For experimental analysis, we used three dataset that sourced from Kaggle. The first dataset comprises 49,582 IMDB movie reviews [12] while the second dataset is Yelp reviews [13] consists of 100,000 reviews. Lastly, we utilized the Trip Advisor reviews dataset [14] which includes 18,307 reviews. These datasets classified into positive and negative sentiments.

2.2. Data Preprocessing

Text preprocessing is an essential step in NLP tasks that involves cleaning and converting unstructured text data to prepare before analyze it. Real-world data generally contain noise, irrelevant characters, formatting issues, wrong spelling, and maybe unnecessary parts that can lower the performance for machine learning models. Hence, text preprocessing needs to be done to make the data suitable for the model, that will improve its performance and efficiency [1], [8]. In this case, text preprocessing model involves lower casing, lemmatization, stemming and removal of HTML tags, links, punctuations, numbers, single characters, and stop words. Lowercase ensures uniformity by converting all text to lowercase. Lemmatization and stemming reduce words to their basic or root forms, capturing the essence of the different inflections. Removal of HTML tags, links, punctuation, numbers, single characters, and stop words helps eliminate distractions and irrelevant information.

2.3. Word2Vec Training

For machine learning model to process a sentence into NLP task, words must be converted into a vector form. This proses is known as word embedding. Word2Vec is one of the famous methods for learning word embeddings as they use shallow neural network for processing a text before passing it into a deep learning algorithm. The embeddings can be obtained using Skip Gram model and Common Bag of words (CBOW) model. The CBOW model predicts the current word from surrounding context words, whereas Skip Gram model predicts the surrounding context words from the current word. During training, the Word2Vec system receives preprocessed dataset as input. The Word2Vec system receives the pre-processed data as input and returns a vector representation of each word as output. The first step in developing a Word2Vec model is to generate a vocabulary from input data. Next, the word's vector representation is learned. The words are mapped into a word matrix and are converted into vectors in an n-dimensional vector space by representing similar words near to each other [5].

2.4. Convloutional Neural Network (CNN)

Convolutional Neural Network (CNN) is artificial neural network specifically designed for processing images. However, the applicability of CNNs extends to 1D data such as text data and has proven effective in NLP tasks [6]. This possible by make the input text is represented as a sequence of words and treat each word as a "pixel" in a grid. To apply a CNN model to 1D input data, the model is structured with several layers that hierarchically extract features from the input text.

Basic CNNs function by feeding multidimensional input (such as images and word embeddings) to a Convolutional layer, which is made up of several filters that learn unique features. Note how these filters are applied subsequently to different areas of the input. In most cases, the output is pooled or subsampled to smaller dimensions before being fed into a connected layer.

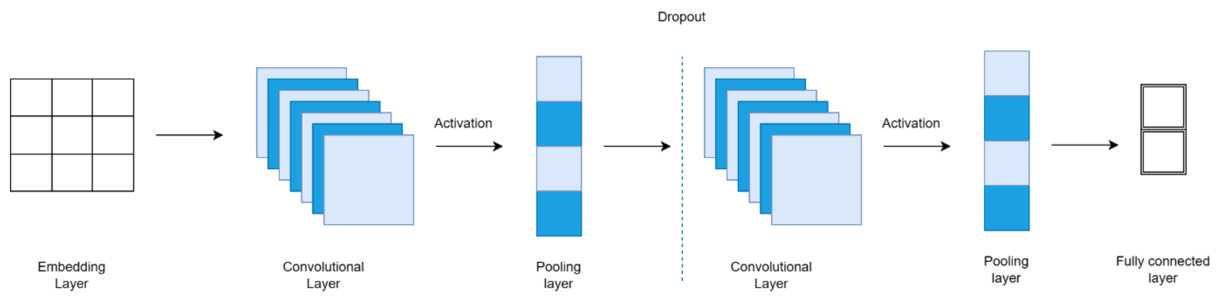

Figure 1. CNN Model with 2 Layer Conv1D

2.5. RNN

Recurrent Neural Networks (RNNs) are ideal neural network model in NLP task including sentiment analysis. Different from traditional forward neural networks where each layer only receives input from the previous layer and lacks feedback connections, making FNN is suitable for task where sequential is not crucial like image processing, RNNs are specifically designed for sequential data. Just like architecture in Figure 2, RNNs have connections forming directed cycles to retain information across different time steps. This feature that makes RNNs highly effective for NLP tasks and time series prediction.

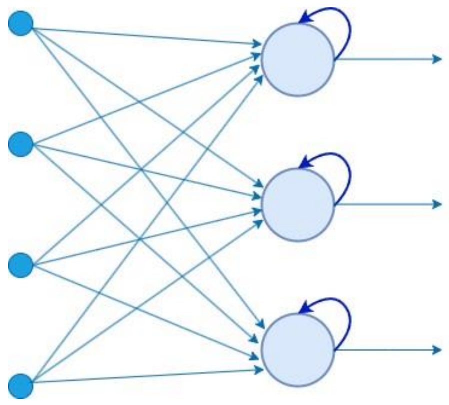

Figure 2. Neural Network with One Recurrent Layer

Handling variable-length sequences presents a challenge for simple RNN due to sequential processing and potential memory degradation. GRU and LSTM can handle this issue with gating and memory cells mechanism. With These features, GRU and LSTM effectively manage variable-length sequences by selectively remember and forgot relevant information, regardless of the length of sequence. This advantage is especially valuable in tasks such as sentiment analysis for large datasets and relatively long sequences. The RNNs model used in this research is shown in Figure 3.

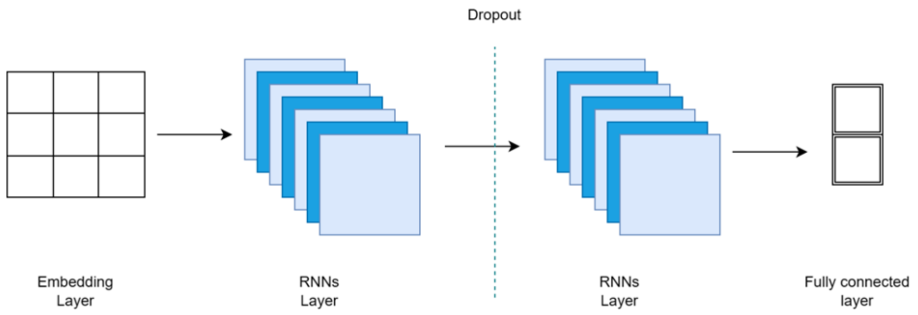

Figure 3. RNNs Model

2.5.1. LSTM

LSTM which stands for Long Short-Term Memory is a version of RNNs that have more complex structure, where each unit has three gates: an input gate, an output gate, and forget gate. The forget gate decides which current value that needs attention and which can be ignored. The input gate performs the following operations to update the cell status. And the output gate determines whether the memory cell should influence the output at the current time step.

2.5.2. GRU

Gated Recurrent Unit is a version of RNNs that is similar with LSTM with a simpler gating mechanism. The main difference between GRU and LSTM is the way they handle the memory cell state. In LSTM, the memory cell state is maintained separately from the hidden state and is updated using three gates: the input gate, output gate, and forget gate. In GRU, the memory cell state is replaced with a candidate activation vector, which is updated using two gates: the reset gate and update gate. The reset gate determines how much of the previous hidden state to forget, while the update gate determines how much of the candidate activation vector to incorporate into the new hidden state.

2.6. CNN-RNNS

The hybrid CNN-RNNs model architecture that used in this research consists of a series of Keras layers designed for text classification tasks. First layer is an embedding layer, which is responsible for converting input indices into dense vector representations to capture semantic meaning. Second layer is a 1D convolution layer to captures local patterns and features of the embedding output. Followed by the Max-Pooling layer to reduces the dimensionality of feature maps and retaining the most important information. The fourth layer is dropout layer to prevent overfitting by randomly deactivating 20% of the neurons during training. Then the output of Max-Pooling layer becomes the input to the LSTM network, which measures the long-term dependencies of the feature sequences. One of the benefits of LSTM is that it can capture long-term dependencies between regions by considering previous data. The outputs of the LSTM are concatenated and fed to a fully connected layer, and an activation function Sigmoid is applied to generate the final output prediction i.e., positive or negative text. The advantage of this model is that the first convolutional layer extracts local features and the LSTM can use the order of these features to learn more about the order of the input text.

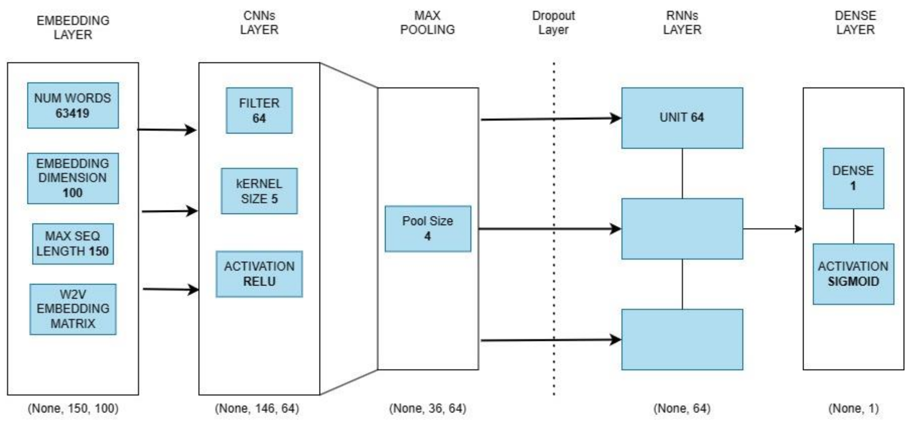

Figure 4. CNN-RNNs Model

2.7. RNNs-CNN

The idea of this model is LSTM layer act as an encoder such that for every token in the input there is an output token that contains information not only of the original token, but all other previous tokens. Afterwards, the CNN layer will find local patterns using this richer representation of the original input. The hybrid model architecture consists of a series of Keras layers and the details are discussed as follows.

First layer is an embedding layer resulting of word embeddings of each token and then fed into RNNs layer. The LSTM continuously updates its internal state at each time step based on the current input and the information stored in the previous state. So, each output time step consists of information not only the current sequence but also the previous sequences. In other words, the LSTM layer is generating a new encoding for the original input. Followed by dropout layer prevent overfitting and enhance generalization. Next layer is an 1D convolutional layer to captures local patterns and features of the output from previous layer. After the Conv1D layer, there is a the Global-Max-Pooling layer to aggregate the most important features from the previous layer. Finally, a dense layer is added with sigmoid activation function to produce the final output probabilities for each class.

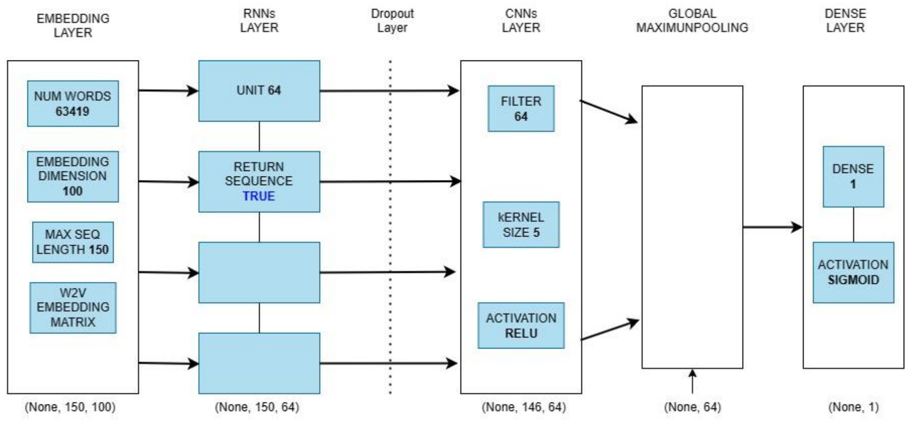

Figure 5. RNNs-CNN Model

3. Results

3.1. Experiment Setup

Table 1. Dataset Parameters

Dataset	Max Seq Length	Batch Size
IMDB Movie (A)	150	512
YELP (B)	120	512
Trip Advisor (C)	200	128

In this experiment, we utilized three distinct datasets: IMDB movie reviews, yelp reviews and trip advisor reviews. The data was partitioned in three different ratios for training and testing: 40% training and 60% testing, 60% training and 40% testing, and 80% training and 20% testing. Additionally, 20% of training data is used for validation during training the model.

3.2. Parameters

Table 2. Model Parameters

Embedding Dimension	100
Epoch	5
Filters, Units	64
Kernel Size	5
Pool Size	4
Dropout	0.2

As such, the neural network layer was standardized to 64 units across all models so to determine how much complexity it takes for models to perform well.

3.3. Model Comparisons

Table 3. Test Accuracy and Training Time for Dataset A

Model Dataset A	40:60		60:40		80:20	
	Test	Time	Test	Time	Test	Time
CNN	0.832	25.4	0.857	35.1	0.866	42.6
LSTM	0.840	28.7	0.842	38.6	0.851	50.5
GRU	0.779	28.4	0.827	40.6	0.835	46.8
CNN-LSTM	0.857	25.3	0.871	33.9	0.884	43.9
CNN-GRU	0.859	24.4	0.873	34.4	0.885	41.9
LSTM-CNN	0.866	26.3	0.878	37.1	0.885	44.6

GRU-CNN	**0.867**	26.6	**0.881**	37.4	**0.887**	44,8

On dataset A, the GRU Model obtained the lowest accuracy results in all three split scenarios. Although it has a slightly longer training time compared to the CNN model, the hybrid model demonstrated superior accuracy in the test, outperforming the basic model in split scenarios. Overall, the GRU-CNN model consistently achieved the highest test accuracy across all data splits. This suggests that combining different model architectures may be more effective in capturing different patterns in the data.

Table 4. Test Accuracy and Training Time for Dataset B

Model Dataset B	40:60		60:40		80:20	
	Test	Time	Test	Time	Test	Time
CNN	0.892	39.3	0.901	50.9	0.904	68.4
LSTM	0.842	46.3	0.890	57.4	0.907	85.0
GRU	0.842	45.6	0.827	57.1	0.888	79.1
CNN-LSTM	0.902	37.9	0.906	53.5	0.913	68.9
CNN-GRU	0.903	42.9	0.907	50.8	0.912	70.1
LSTM-CNN	0.903	42.3	**0.912**	53.9	**0.916**	74.1
GRU-CNN	**0.906**	42.2	0.911	52.2	0.915	73.2

On dataset B, the accuracy results from almost all models achieved improvement in each split dataset. Despite required longer training time, the basic GRU model achieved the lowest accuracy test. Meanwhile, the hybrid model once again outperformed the individual models with slightly longer training time than CNN model. The hybrid LSTM-CNN model displayed the highest result with a 91.6% accuracy test. Based on increased accuracy across dataset splits and competitive training times, the hybrid models display their ability to learn more complex patterns and relationships from data.

Table 5. Test Accuracy and Training Time for Dataset C

Model Dataset C	40:60		60:40		80:20	
	Test	Time	Test	Time	Test	Time
CNN	0.926	30.1	0.936	41.7	0.938	46.2
LSTM	0.834	43.9	0.865	53.0	0.889	56.9
GRU	0.826	37.3	0.827	49.1	0.862	56.3
CNN-LSTM	0.939	33.8	0.946	42.8	0.951	51.8
CNN-GRU	0.931	30.8	0.949	42.8	0.956	48.6
LSTM-CNN	0.948	33.7	0.950	42.8	0.951	52.4
GRU-CNN	0.946	33.3	0.954	41.4	0.957	55.5

Similar to datasets A and B, the best accuracy of the deep learning model trained with dataset c was achieved by the hybrid model. Notably, the accuracy of RNNs-CNN models results in the 40:60 dataset split outperformed the individual model results that in all split scenarios. With the GRU-CNN model topped the results with a test accuracy of 0.957 and a training time of 55.5 seconds. And CNN model achieved the highest accuracy among the individual models with a 93.8% accuracy and the shortest training time of 46.2 seconds. This suggest that the hybrid model had the ability to capture both short-term and long-term dependencies and local feature in the data, thereby resulting better accuracy result.

3.4. Further Results

3.4.1. Units

Table 6. Test Results from the GRU-CNN Model with Different Units Values

Units	Accuracy	F1 Score	Precision	Recall	Time Training
256	**0.961**	0.961	0.960	0.961	91.4
128	0.961	0.961	0.961	0.961	79.3

64	0.955	0.954	0.954	0.955	75.1

The GRU-CNN model's performance was trained on dataset C with an 80: 20 splits and unit values differed as listed in Table 4.6. The model provided accuracy of 0.961, F1 score of 0.961, precision of 0.960 and recall of 0.961 with respect to 256 units and the time for training was 91.4. Likewise, for 128 units of the model it showed excellent results, with accuracy of 0.961, F1 score, precision, and recall of 0.961 reducing the training time to 79.3. Afterwards, the units were reduced further to 64 and slightly reduced efficiency was noticed for the same metrics - accuracy, F1-score, precision and recall as 0.955 and training time 75.1. This shows that GRU-CNN model performed well in all the units' values with negligible variance in the resulting metrics.

3.4.2. Optimizer

Table 7. Test Results from the GRU-CNN Model with Different Optimizers

Optimizer	Accuracy	F1 Score	Precision	Recall
Adam	0.961	0.957	0.957	0.957
SGD	0.947	0.947	0.947	0.947
RMSprop	0.955	0.955	0.955	0.955

Among various optimizers, the best results were obtained by Adams optimizer with 96% accuracy, F-score, and precision, recall all being at 96 as shown in Table 4.7%. Coming close behind was the SGD optimizer with good overall indicators with an accuracy, F1 score, precision, and recall of 0.947. Another optimizer is RMSprop achieved a result with accuracy, f1 score, precision, and recall of 0.955. Finally, all these results seem to emphasize that the type of optimizer makes a tremendous difference in the model performances and performance, wherein Adam has better metrics than SGD and RMSprop.

3.4.3. Learning Rates

Table 8. Test Results from the GRU-CNN Model with Different Learning rates

Learning Rate	Accuracy	F1 Score	Precision	Recall
0.01	0.951	0.951	0.951	0.951
0.001	0.962	0.962	0.962	0.962

0.0001	0.952	0.949	0.949	0.949
0.00001	0.836	0.774	0.842	0.836

The hybrid GRU-CNN model performance also evaluated with different learn rates as it is seen in table 4.8. The highest accuracy, F1 score, precision and recall were achieved by learning rate of 0.001 with value all being 95,1%. However, a learning rate of 0.00001 resulted in much poorer metrics that showed a decline in performance as accuracy was 0.836, F1-score of 0.774, precision of 0.842, and recall of 0.836. The learning rates of 0.01 and 0.0001 showed an intermediate performance with accuracy of 0.951 and 0.952 respectively. The findings underscore the importance of the learning rate on an improved GRU-CNN for overall better performance with a value of 0.001 being optimal.

4. Conclusion

Based on the conducted experiments, it can be concluded that By combining the advantages of CNN which is able to recognize local patterns and RNN's ability to learn sequences in text, it can produce better accuracy. Consistent accuracy results demonstrate that hybrid models consistently outperform basic models. Although hybrid models generally require slightly more training time, this trade-off is tolerable and justified for tasks where accuracy is crucial.

Evaluations of the GRU-CNN model on Dataset C reveal that different unit values minimally impact model’s accuracy, while the choice of optimizer and learning rate significantly differs. Therefore, for sentiment analysis tasks, the hybrid GRU-CNN architecture is a promising choice for achieving higher accuracy even with low computation resources.

5. Future Work

For future research, it is recommended to explore different hybrid approaches, such as parallel deep learning hybrid models and combinations of deep learning and machine learning, across various datasets. This broader exploration can provide insights into the generalizability and versatility of hybrid models in different contexts and further enhance their applicability in sentiment analysis and related tasks.

References

[1] V. V. Nhlabano and P. E. N. Lutu, "Impact of Text Pre-Processing on the Performance of Sentiment Analysis Models for Social Media Data," 2018 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD), Durban, South Africa, 2018, pp. 1-6, doi: 10.1109/ICABCD.2018.8465135.

[2] Rahman and M. S. Hossen, "Sentiment Analysis on Movie Review Data Using Machine Learning Approach," 2019 International Conference on Bangla Speech and Language

Processing (ICBSLP), Sylhet, Bangladesh, 2019, pp. 1-4, doi: 10.1109/ICBSLP47725.2019.201470.

[3] K. K. Agustiningsih, E. Utami, and M. A. Alsyaibani, “Sentiment Analysis of COVID-19 Vaccines in Indonesia on Twitter Using Pre-Trained and Self-Training Word Embeddings,” Jurnal Ilmu Komputer dan Informasi, vol. 15, no. 1, pp. 39-46, Feb. 2022.

[4] E. M. Dharma, F. L. Gaol, H. L. H. S. Warnars, and B. Soewito, "The Accuracy Comparison Among Word2Vec, Glove, and FastText Towards Convolution Neural Network (CNN) Text Classification," Journal of Theoretical and Applied Information Technology, vol. 100, no. 02, pp. 350-359, Jan. 31, 2022.

[5] R. Catelli, S. Pelosi, and M. Esposito, "Lexicon-Based vs. Bert-Based Sentiment Analysis: A Comparative Study in Italian," Electronics, vol. 11, no. 3, art. no. 374, Jan. 2022. doi: 10.3390/electronics11030374.

[6] R. Naquitasia, D. H. Fudholi, and L. Iswari, "Analisis Sentimen Berbasis Aspek pada Wisata Halal dengan Metode Deep Learning," JTI, vol. 16, no. 2, pp. 156, Jul. 2022, doi: 10.33365/jti.v16i2.1516.

[7] R. Kusnadi, Y. Yusuf, A. Andriantony, R. Ardian Yaputra, and M. Caintan, "Analisis Sentimen Terhadap Game Genshin Impact Menggunakan BERT," Rabit, vol. 6, no. 2, pp. 122-129, Jul. 2021.

[8] Z. Jianqiang and G. Xiaolin, "Comparison Research on Text Pre-processing Methods on Twitter Sentiment Analysis," in IEEE Access, vol. 5, pp. 2870-2879, 2017, doi: 10.1109/ACCESS.2017.2672677.

[9] W. Widayat, "Analisis Sentimen Movie Review menggunakan Word2Vec dan metode LSTM Deep Learning," J. Media Informatika Budidarma, vol. 5, no. 3, pp. 1018-1026, Jul. 2021. doi: 10.30865/mib.v5i3.3111.

[10] P. Jain, V. Saravanan, and R. Pamula, "A Hybrid CNN-LSTM: A Deep Learning Approach for Consumer Sentiment Analysis Using Qualitative User-Generated Contents," ACM Transactions on Asian and Low-Resource Language Information Processing, vol. 20, pp. 1-15, Sep. 2021. doi: 10.1145/3457206.

[11] D. Hermanto, A. Setyanto, and E. Luthfi, "Algoritma LSTM-CNN untuk Binary Klasifikasi dengan Word2vec pada Media Online," Creative Information Technology Journal, vol. 8, pp. 64, Mar. 31, 2021. doi: 10.24076/citec.2021v8i1.264

[12] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning Word Vectors for Sentiment Analysis," in Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA, June 2011, pp. 142-150, [Online]. Available: http://www.aclweb.org/anthology/P11-1015.

[13] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in Advances in Neural Information Processing Systems, vol. 28, 2015.neurips.cc/paper_files/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf

[14] Barkha Bansal, “TripAdvisor Hotel Review Dataset”. Zenodo, Apr. 17, 2018. doi: 10.5281/zenodo.1219899.