Accredited by National Journal Accreditation (ARJUNA) Managed by
Ministry of Research, Technology, and Higher Education, Republic Indonesia with Second Grade (Peringkat 2, Sinta 2)
since year 2017 to 2021 according to the decree No. 10/E/KPT/2019.

Published online on the journal's webpage: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 5 No. 5 (2021) 1001 - 1007 ISSN Media Elektronik: 2580-0760

Improving Accuracy using The ASERLU layer in CNN-BiLSTM Architecture on Sentiment Analysis

Sandi Hermawan[1], Rilla Mandala[2]
[1]Master of Science in Information Technology, Faculty of Computing, President University
[2]School of Electrical Engineering and Informatics, Bandung Institute of Technology
[1]sandi.hermawan@student.president.ac.id, [2]rila@informatika.org

Abstract

There have been 350,000 tweets generated by the interaction of social networks with different cultures and educational backgrounds in the last ten years. Various sentiments are expressed in the user comments, from support to hatred. The sentiments regarded the United States General Election in 2020. This dataset has 3,000 data gotten from previous research. We augment it becomes 15,000 data to facilitate training and increase the required data. Sentiment detection is carried out using the CNN-BiLSTM architecture. It is chosen because CNN can filter essential words, and BiLSTM can remember memory in two directions. By utilizing both, the training process becomes maximum. However, this method has disadvantages in the activation. The drawback of the existing activation method, i.e., "Zero-hard Rectifier" and "ReLU Dropout" problem to become the cause of training stopped in the ReLU activation, and the exponential function cannot be set become the activation function still rigid towards output value in the SERLU activation. To overcome this problem, we propose a novel activation method to repair activation in CNN-BiLSTM architecture. It is namely the ASERLU activation function. It can adjust positive value output, negative value output, and exponential value by the setter variables. So, it adapts more conveniently to the output value and becomes a flexible activation function because it can be increased and decreased as needed. It is the first research applied in architecture. Compared with ReLU and SERLU, our proposed method gives higher accuracy based on the experiment results.

Keywords: CNN-BiLSTM, ReLU, Sentiment Analysis, SERLU, US Election 2020.

1. Introduction

In the last ten years, around 350,000 tweets are generated by social network interaction with different cultures and educational backgrounds. The freedom of speech with anonymous users can be using support or criticism sentences. Support sentence is expressed in good words to get a positive image from the political party figure. Therefore, it can affect everyone that the figure is worthy or deserving of a good predicate. Meanwhile, criticizing sentence is expressed in reproachful words to get a negative image from that figure. It can affect everyone that the figure is not worthy because it has a lousy predicate [1].

Twitter users expressed various sentiments in the United States general election in 2020. From excessive support to dirty words written on Twitter social media. Fanatical Twitter users will always support their favorite figure and provoke the figure's enemy. However, passive Twitter users always reveal the reality according to the conditions at that time. Therefore, it is exciting to examine more deeply to find out the feeling expressed by the users [2].

Sentiment analysis in tweets is carried out using a deep learning method to produce the best performance. Several studies were conducted to detect these sentiments, i.e., Convolutional Neural Network (CNN) [3], Bidirectional Long Short Term Memory (BiLSTM) [4], and Convolutional Neural Network - Bidirectional Long Short Term Memory (CNN-BiLSTM) [5]. Eventually, CNN-BiLSTM produces the best accuracy among several methods carried out.

Although the CNN-BiLSTM architecture is the best method, some drawbacks must be overcome in this architecture. Because according to Qiu et al. [6], the

Accepted by editor: 30-09-2021 | Final Revision: 29-10-2021 | Online Publication: 31-10-2021

ReLU (Rectifier Linear Unit) activation function in the CNN architecture has a "Zero-hard Rectifier" factor which causes the network connected to it will lose the benefit of negative values so that the accuracy achieved is not optimal. Likewise, Parisi et al. [7] argue that the "ReLU dropout" problem causes training to be disrupted, where the weights should get negative value are forced to become the value of 0 for each negative weight.

To overcome the ReLU problem, a study conducted by Ashiquzzaman et al. [8] proposed the ELU (Exponential Linear Units) activation function to recognize compound characters in Bangla language using the Convolutional Neural Network (CNN) and the efficient layer training approach with the dropout method, which is believed to replace ReLU activation. In this study, the dropout method and the proposed ELU activation function were applied to reduce data overfitting and improve compatibility words in the prediction model. The method tested in the study were SVM (Support Vector Machine), CNN (ReLU layered), and CNN (ELU layered and Dropout). The CNN (ELU layered and Dropout) method is higher than the SVM and CNN (ReLU layered). However, the ELU has a disadvantage in that it cannot adjust the positive value.

A study conducted by Zhang and Li [9] proposed a new activation function, namely Scaled Exponentially Regularized Linear Units (SERLU), to increase the accuracy achieved on activation. It is also to fix the ELU activation issue. The data used in the study were CIFAR10, CIFAR 100, and MNIST have been augmented. The activation functions tested in the study were SERLU, SELU, ELU, Swish, Leaky ReLU, and ReLU. As a result, SERLU becomes better activation than the other activation was experimented with in various datasets.

Although the SERLU activation function is considered the best in state-of-the-art method, it needs improvement because it is still too rigid when used in the architecture. After all, it cannot be flexible with positive and negative values. Therefore, it is necessary to adjust on activation function so that the expected output value from the prediction can increase the accuracy value more maximally. As far as we know, there are no research on how to improve accuracy to become better through SERLU activation. For this reason, we propose a method to repair it and overcome the ReLU problem. The activation function is called Adjustable Scaled Exponentially-Linear Units (ASERLU), resulting from modified SERLU activation [9].

More related works are discussed in section 2, while more information on the proposed method is discussed in section 3. Section 4 describes the experiment and evaluation, and the conclusion and future work are presented in section 5.

2. Research Method

In our study, we propose the research flow to explain the experiment process, i.e., word contraction, data cleaning, case folding, easy data augmentation process, the second stage of word contraction, the second stage of data cleaning, the second stage of case folding, word stemming, lemmatization, stopword filtering, dataset splitting, tokenizing, encoding, padding, dataset partition (data training, validation, and testing), Cross-Validation, and evaluation. The research flow of our study is shown in Figure 1.

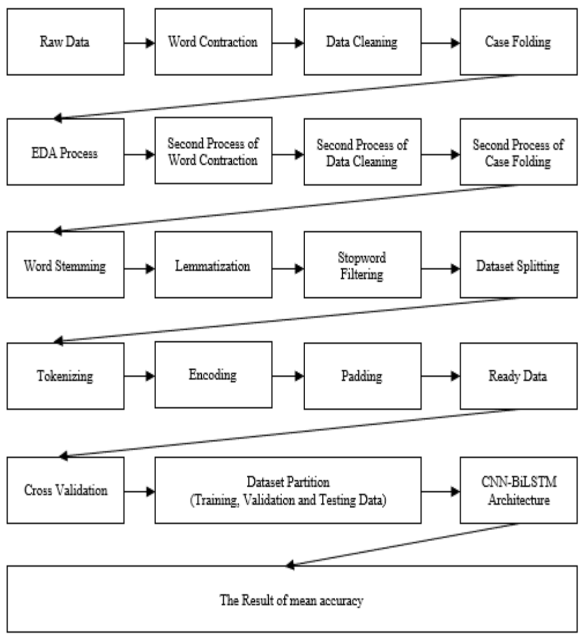

Figure 1. The proposed research flow is carried out in our study

In our proposed, the CNN-BiLSTM architecture is arranged from the input layer to the output layer, as shown in Figure 2 and Table 1.

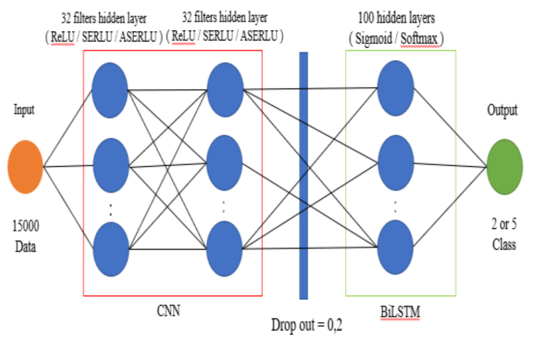
32 filters hidden layer (ReLU / SERLU / ASERLU)
32 filters hidden layer (ReLU / SERLU / ASERLU)
100 hidden layers (Sigmoid / Softmax)
Input
15000 Data
Output
2 or 5 Class
CNN
Drop out = 0,2
BiLSTM

Figure 2. The CNN-BiLSTM architecture is used in our study

The ReLU activation is better than Sigmoid because ReLU can significantly reduce the loss in the first or middle hidden layer [10]. Sigmoid is not suitable to be placed in this hidden layer because its characteristic is always trapped at the minimum local gradient, so the training takes a long time to predict the data. ReLU removes the negative weight on the architecture to speed up data recognition training. It is more effective than Sigmoid. Therefore, ReLU is widely used because it has fast learning. The ReLU activation formula is shown in Table 2 and Figure 3.

Table 1. The Detail of CNN-BILSTM Architecture in Application

Layer Part	Explanation
Input Layer	15,000 data
CNN Architecture	30 hidden layers (activation of ReLU, SERLU, and ASERLU)
1 Dimension Max pooling	*pool_size* = 2
CNN Architecture	30 hidden layers (activation of ReLU, SERLU, and ASERLU)
1 Dimension Max pooling	*pool_size* = 2
BiLSTM Architecture	100 hidden layers with *dropout* and *recurrent_dropout* are 0,2
Dense	Sigmoid for binary class or Softmax for multi class
Output Layer	2 or 5 class

Table 2. Activation Function owned by The RELU Layer

Activation	Output Value	Condition
ReLU(x)	x	if $x > 0$
	0	if $x \leq 0$

where x is the input value that will go into activation.

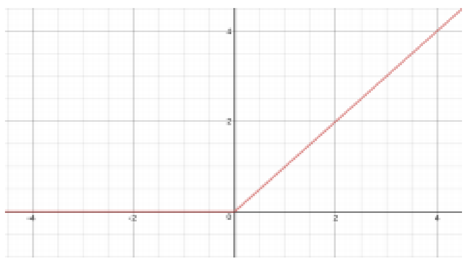

Figure 3. ReLU activation function which is illustrated in the graph [11]

The SERLU activation can repair ReLU because ReLU has potential backpropagation that has been generated from the gradient is too small [9]. It happens because ReLU activation has the output value of 0 if $x < 0$ and x if $x > 0$. This problem is often are called vanishing gradients [12]. When the network does not get the score, the weight cannot set its value and the learning process will be stopping. The ReLU effect propagates to the associated hidden layer. It is called dying ReLU [13]. Eventually, Leaky ReLU fixed this deficiency. This activation improves ReLU so that there is no vanishing gradient. The Leaky ReLU activation value is x if $x > 0$ and $0.01x$ if $x < 0$ [14]. This method is also insufficient to improve the resulting gradient because it cannot control the negative value well. So, the Scaled Exponential Linear Units (SELU) is introduced with a normalization approach on the architecture [15]. The SELU activation is further developed into SERLU Activation. The SERLU activation formula is shown in Table 3 and Figure 4.

Table 3. Activation Functions Owned By The SERLU Layer

Activation	Output Value	Condition
SERLU(x)	$\lambda_{serlu} \,.\, x$	if $x \geq 0$
	$\lambda_{serlu} \,.\, \alpha_{serlu} \,.\, x \,.\, (e^x)$	if $x < 0$

where x is the input value that will go into activation, λ_{serlu} (*lambda of SERLU*) is the scale value contained in the activation has a value of 1.07862, α_{serlu} (*alpha of SERLU*) is the regulation value contained in the activation value of 2.90427 from the negative input x, and e is the natural constant value (*Eulerian value*)

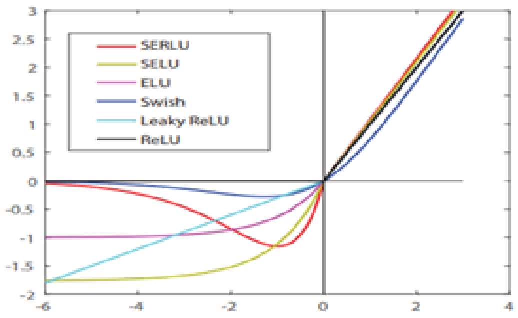

Figure 4. Graph generated by SERLU activation with other activations [9]

Based on the problem discussed in the introduction, we propose a novel method to repair activation in CNN-BiLSTM architecture, namely ASERLU. The ASERLU activation is a new activation adopted from a modified SERLU activation so that it is easy to adapt with output value [9]. It can also cover the shortcoming of SERLU, which cannot set exponential value. The advantage of ASERLU activation is that there are settings provided by *control positive* (*cp*), *first control negative* (cn_1), and *second negative control* (cn_2) that can be adjusted with the desired value. So, the setter variables in ASERLU activation can flexible towards the negative and positive output values and increase the prediction results to be more precise. The ASERLU activation formula is shown in Table 4.

Table 4. Activation Function owned by The Proposed Activation Method

Activation	Output Value	Condition
ASERLU(x)	$\lambda_{serlu} \,.\, x \,.\, cp$	if $x \geq 0$
	$\lambda_{serlu} \,.\, \alpha_{serlu}\, (cn_1 \,.\, x)\, (e^{x \,.\, cn_2})$	if $x < 0$

where x is the input value that will go into activation, λ_{serlu} (*lambda of SERLU*) is the scale value contained in the activation has a value of 1.07862, α_{serlu} (*alpha of SERLU*) is the regulation value contained in the activation value of 2.90427 from the negative input x, e is the natural constant value (*Eulerian value*), *cp* (*Control Positive*) is the setter variable for positive x input (free to set), cn_1 (*First Control Negative*) is the first setter variable for negative x input (free to set), and cn_2 (*Second Control Negative*) is the second setter variable for negative x input (free to set).

According to Wei and Zou (2019) [14], "Data augmentation is randomizing text elements to create the new text.". For example, each word is slightly changed by creating a new sentence to get much data. This option only applies to the classification algorithm that does not take word order into the sentence. In practice, every sentence must be made into words. Then, the augmentation process is carried out and recombines to make new sentences. The new data approach is carried out in 4 ways, i.e. Synonym Replacement (SR) is a way to choose *n* word in sentences that not the stopword, Random Insertion (RI) is a way to find random word synchronization in the sentence that is not the stopword, Random Swap (RS) is a way to randomly select two words in a sentence and swap their positions. To be able to do this according to *n* times and Random Deletion (RD) is a way to randomly delete every word in a sentence with probability *p*.

In use, the EDA theory varies the changed word amount, i.e., *n* for SR, RI, and RS, based on the sentence length denoted by *l* written in formula 2.

$$n = \alpha \cdot l \qquad (2)$$

where α is a parameter that shows the percent of the words in a sentence are changed (this theory uses $p = \alpha$ for RD), *n* is the amount of variation word changed in the sentence, and *l* is the sentence length.

3. Result and Discussion

The data used in our study is data taken from the University of Stuttgart's website, containing 3,000 data conducted by Grimminger and Klinger [2]. The data contains columns, i.e., 'text' contains user comments, 'trump' contains sentiments about the US presidential candidate of Donald Trump, 'biden' contains sentiments about the US presidential candidate of Joe Biden, 'west' contains sentiments from westerners in general, and 'HOF' contains the sentiments from previous researchers directly. There are five classes on the 'trump' and 'biden' label, i.e., Against means expressing the word hate, Neither means not expressing the word hate but expressing support for the political figure's enemy, Neutral mention means not expressing the word support or hatred, Mixed means expressing the word support but offending the figure and Favor means providing support to that figure. On the 'trump' label, Against contains 842 data, Neither contains 1,017 data, Neutral mention contains 341 data, Mixed contains 20 data, and Favor contains 780 data. On the 'biden' label, Against contains 404 data, Neither contains 404 data, Neutral mention contains 326 data, Mixed contains 47 data, and Favor contains 1,236 data. The 'west' label is not included in the data because it does not meet the criteria because there is 1 class, namely neither. There are two classes on the 'hof' label, i.e., Non-hateful means that the sentiment does not contain elements of hatred and offensive words, and Hateful means that the sentiment contains elements of hatred and offensive words. On the label 'hof', Non-hateful contains 2,648 data, and Hateful contains 352 data.

We augment it becomes 15,000 *data*. The activations are compared by knowing the test results carried out. The best mean accuracy among them is selected. The test was carried out by three activation methods, i.e., ReLU, SERLU, and ASERLU. It uses two times Cross-Validation with *k* = 10 (*k* is the number to divide each fold on the dataset). The comparison of the data used in our study was 90% training data (13,500 data), 5% validation data (750 data), and 5% testing data (750 data). Tests are carried out alternately from the first fold to the tenth fold. The data is randomized with *random_state* = 0. The calculation of each model use *optimizer* = "Adam", *loss* = "categorical_crossentropy", and *epoch* = 10.

In the dataset partition, the best dataset comprises three parts, i.e., training, validation, and testing data. Training data are trained to build the formed model, validation data evaluate the training model periodically, and testing data evaluate the final results obtained after the model has been completed to train and validate data. Process for training, validation, and testing data must be used in Cross-Validation on each experiment [15]. We propose that Cross-Validation is carried out two times in our study, i.e., Cross-Validation Part A and Cross-Validation Part B. cross-validation Part A (the first 750 data for data validation and the second 750 data for data testing) and cross-validation Part B (the first 750 data for data testing and the second 750 data for data validation). The result of mean accuracy for Part A and B is calculated from 10 folds of each Cross-Validation. The final calculation of mean accuracy for all parts can be obtained from the mean accuracy of Part A and B. The following is the arrangement of distribution for Cross-Validation Part A and B in Table 5.

Table 5. The Proposed Cross-Validation Applied To CNN-BiLSTM Architecture

Part	Fold	Training Set	Validation Set	Testing Set
A	1	1,500th until 15,000th data	1st until 750th data	750th until 1,500th data
	2	1st until 1,500th and 3,000th until 15,000th data	1,500th until 2,250th data	2,250th until 3,000th data
	3	1st until 3,000th and 4,500th until 15,000th data	3,000th until 3,750th data	3,750th until 4,500th data
	4	1st until 4,500th and 6,000th until 15,000th data	4,500th until 5,250th data	5,250th until 6,000th data
	5	1st until 6,000th and	6,000th until 6,750th data	6,750th until 7,500th data

		7,500th until 15,000th data		
	6	1st until 7,500th and 9,000th until 15,000th	7,500th until 8,250th	8,250th until 9,000th
	7	1st until 9,000th and 10,500th until 15,000th data	9,000th until 9,750th data	9,750th until 10,500th data
	8	1st until 10,500th and 12,000th until 15,000th data	10,500th until 11,250th data	11,250th until 12,000th data
	9	1st until 12,000th and 13,500 until 15,000 data	12,000th until 12,750th data	12,750th until 13,500th data
	10	1st until 13,500th data	13,500th until 14,250th data	14,250th until 15,000th
B	1	1,500th until 15,000th data	750th until 1,500th data	1st until 750th data
	2	1st until 1,500th and 3,000th until 15,000th data	2,250th until 3,000th data	1,500th until 2,250th data
	3	1st until 3,000th and 4,500th until 15,000th data	3,750th until 4,500th data	3,000th until 3,750th data
	4	1st until 4,500th and 6,000th until 15,000th data	5,250th until 6,000th data	4,500th until 5,250th data
	5	1st until 6,000th and 7,500th until 15,000th data	6,750th until 7,500th data	6,000th until 6,750th data
	6	1st until 7,500th and 9,000th until 15,000th	8,250th until 9,000th	7,500th until 8,250th
	7	1st until 9,000th and 10,500th until 15,000th data	9,750th until 10,500th data	9,000th until 9,750th data
	8	1st until 10,500th and 12,000th until 15,000th data	11,250th until 12,000th data	10,500th until 11,250th data
	9	1st until 12,000th and 13,500 until 15,000 data	12,750th until 13,500th data	12,000th until 12,750th data
	10	1st until 13,500th data	14,250th until 15,000th	13,500th until 14,250th data

So, to calculate all results on mean accuracy shown in formula 3 [15].

$$Accuracy_{ap} = (Accuracy_{cvpa} + Accuracy_{cvpb})/2 \qquad (3)$$

where $Accuracy_{ap}$ is the mean accuracy from all parts are generated by calculation, $Accuracy_{cvpa}$ is the mean accuracy of part A, and $Accuracy_{cvpb}$ is the mean accuracy of part B.

The activations are compared by knowing the test results carried out. The best mean accuracy among them is selected. The test was carried out by three activation methods, i.e., ReLU, SERLU, and ASERLU. It uses two times Cross-Validation with k = 10 (k is the number to divide each fold on the dataset). The comparison of the data used in our study was 90% training data (13,500 data), 5% validation data (750 data), and 5% testing data (750 data). Tests are carried out alternately from the first fold to the tenth fold. The data is randomized with *random_state* = 0. The calculation of each model use *optimizer* = "Adam", *loss* = "categorical_crossentropy", and *epoch* = 10.

Based on the experiment results in Table 6, ASERLU layered CNN-BiLSTM is better than the ReLU and SERLU layers in the experiment. The results are shown by accuracy measurement in all experiments.

Table 6. Accuracy Results are generated on The Trump Dataset using The CNN-BILSTM Architecture with each layer

Dataset	Testing Method	Accuracy	Loss
Trump	ASERLU layered CNN-BiLSTM (cp = 1.2, cn_1 = 1.2 and cn_2 = –1)	**98.35%**	**0.052**
	SERLU layered CNN-BiLSTM	97.97%	0.065
	ReLU layered CNN-BiLSTM	97.21%	0.093
Biden	ASERLU layered CNN-BiLSTM (cp = 1.3, cn_1 = 1.2 and cn_2 = –1.2)	**99.73%**	**0.01**
	SERLU layered CNN-BiLSTM	99.70%	0.01
	ReLU layered CNN-BiLSTM	99.59%	0.015
HOF	ASERLU layered CNN-BiLSTM (cp = 1.3, cn_1 = 1.2 and cn_2 = –1.2)	**99.73%**	**0.01**
	SERLU layered CNN-BiLSTM	99.70%	0.01
	ReLU layered CNN-BiLSTM	99.59%	0.015

Based on several experiments that have been done, the result was bad if the adjustment on ASERLU layered CNN-BiLSTM is made carelessly. In that case, its accuracy value obtained will drop drastically below ReLU layered CNN-BiLSTM, and the learning stops because there is a *NaN* value in the loss, as shown in Table 7.

Table 7. Comparison on Mean Accuracy Results from Several Architectures in The HOF Dataset with Random ASERLU Activation Settings

HOF Dataset Testing Method	Accuracy	Loss
SERLU layered CNN-BiLSTM	99.70%	0.01
ReLU layered CNN-BiLSTM	99.59%	0.015
ASERLU layered CNN-BiLSTM (cp = 0.1, cn_1 = 0.001, and cn_2 = –3)	88.27%	*NaN*

where *NaN* (*Not a Number*) is the infinite number (∞) or divided by 0.

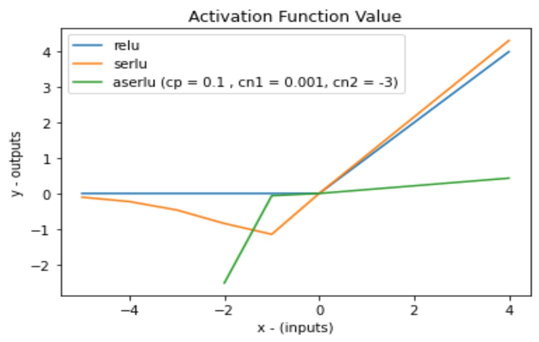

Figure 5. Comparison between ASERLU activation function by cp = 0.1, cn_1 = 0.001 and cn_2 = –3 in variable settings with other activations

Therefore, the activation settings must be adjusted to find the best accuracy. We try to set the cp, cn_1, and cn_2 variables in another number. We use cn_2 variable with positive number but it cannot be maximum, as shown in Table 8.

Table 8. Comparison ASERLU Activation Function use Positive Number cn_2 is not maximized in The HOF Dataset

HOF Dataset Testing Method	Accuracy	Loss
SERLU layered CNN-BiLSTM	99.70%	0.01
ASERLU layered CNN-BiLSTM (cp = 0.9, cn_1 = 0.1, and cn_2 = 0.1)	99.69%	0.011
ReLU layered CNN-BiLSTM	99.59%	0.015

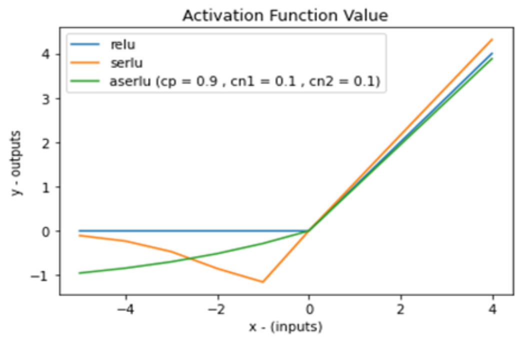

Figure 6. Comparison between ASERLU activation function use pattern by cp = 0.9, cn_1 = 0.1, and cn_2 = 0.1 in variable settings with other activation functions

Because positive number cn_2 variable was failed so that we use negative number cn_2 variable. The activation settings must be adjusted more again to find the best accuracy. We try to set the cp, cn_1, and cn_2 variables in another number again. We use cn_2 variable with negative number but it cannot be maximum, as shown in Table 9.

Table 9. Comparison ASERLU Activation Function use Negative Number cn_2 is not maximized in The HOF Dataset

HOF Dataset Testing Method	Accuracy	Loss
SERLU layered CNN-BiLSTM	99.70%	0.01
ASERLU layered CNN-BiLSTM (cp = 1.2, cn_1 = 1.2 and cn_2 = –1)	99.66%	0.012
ReLU layered CNN-BiLSTM	99.59%	0.015

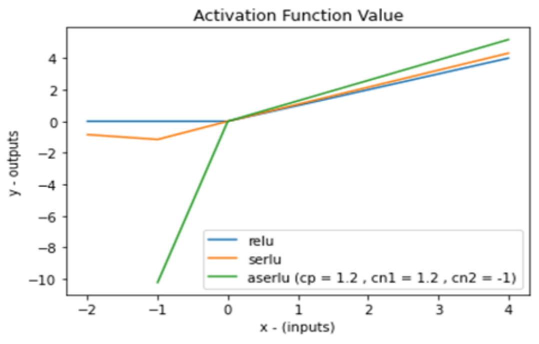

Figure 7. Comparison between ASERLU activation function by cp = 1.2, cn_1 = 1.2, and cn_2 = –1 in variable settings with other activations applied to the HOF dataset

Based on reason with experiment evidence, the ASERLU activation function settings must be set with the suitable variable values to get the maximum result and not stop during data learning.

4. Conclusion

The ASERLU layered CNN-BiLSTM architecture is better than SERLU and ReLU layered CNN-BiLSTM architecture on several datasets that have been carried out in the accuracy measurement. It happens because the ASERLU activation function can adapt toward the predicted data. The setter variables owned by the ASERLU activation function are very influential. It can improve the prediction result and indirectly increase the accuracy initially generated from the ReLU and SERLU activation function if the setter variables are owned ASERLU layered CNN-BiLSTM can be set precisely. However, if the setter variables are set incorrectly, the ASERLU accuracy result is not maximized or even decreases from the ReLU or SERLU activation function. Therefore, The activation settings are significant in this case.

Acknowledgements

I am very grateful to the lecturers of President University who have taught their knowledge so that this research can be achieved.

References

[1] A. Gaydhani, V. Doma, S. Kendre, and L. Bhagwat, “Detecting hate speech and offensive language on twitter using machine learning: An N-gram and TFIDF based approach,” *arXiv*. 2018. https://arxiv.org/pdf/1809.08651.

[2] L. Grimminger and R. Klinger, “Hate Towards the Political Opponent: A Twitter Corpus Study of the 2020 US Elections on the Basis of Offensive Speech and Stance Detection,” 2021. https://arxiv.org/abs/2103.01664.

[3] R. Alshalan and H. Al-Khalifa, “A deep learning approach for automatic hate speech detection in the saudi twittersphere,” *Appl. Sci.*, 2020. https://doi.org/10.3390/app10238614.

[4] H. Mohaouchane, A. Mourhir, and N. S. Nikolov, “Detecting Offensive Language on Arabic Social Media Using Deep Learning,” in *2019 6th International Conference on Social Networks Analysis, Management and Security, SNAMS 2019*,

2019. https://www.doi.org/10.1109/SNAMS.2019.8931839.

[5] I. Abu Farha and W. Magdy, "Multitask Learning for {A}rabic Offensive Language and Hate-Speech Detection," *Proc. 4th Work. Open-Source Arab. Corpora Process. Tools, with a Shar. Task Offensive Lang. Detect.*, 2020. https://aclanthology.org/2020.osact-1.14.

[6] S. Qiu, X. Xu, and B. Cai, "FReLU: Flexible Rectified Linear Units for Improving Convolutional Neural Networks," in *Proceedings - International Conference on Pattern Recognition*, 2018. https://www.doi.org/10.1109/ICPR.2018.8546022.

[7] L. Parisi, D. Neagu, R. Ma, and F. Campean, "QReLU and m-QReLU: Two novel quantum activation functions to aid medical diagnostics," *arXiv*, 2020. https://orcid.org/0000-0002-5865-8708.

[8] A. Ashiquzzaman, A. K. Tushar, S. Dutta, and F. Mohsin, "An efficient method for improving classification accuracy of handwritten Bangla compound characters using DCNN withdropout and ELU," in *Proceedings - 2017 3rd IEEE International Conference on Research in Computational Intelligence and Communication Networks, ICRCICN 2017*, 2017. https://www.doi.org/10.1109/ICRCICN.2017.8234497.

[9] G. Zhang and H. Li, "Effectiveness of scaled exponentially-regularized linear units (SERLUs)," *arXiv*. 2018. https://arxiv.org/pdf/1807.10117.

[10] K. Hara, D. Saito, and H. Shouno, "Analysis of function of rectified linear unit used in deep learning," in *Proceedings of the International Joint Conference on Neural Networks*, 2015. https://www.doi.org/110.1109/IJCNN.2015.7280578.

[11] A. F. M. Agarap, "Deep Learning using Rectified Linear Units (ReLU)," *arXiv*. 2018. https://arxiv.org/pdf/1803.08375.

[12] X. Hu, P. Niu, J. Wang, and X. Zhang, "A Dynamic Rectified Linear Activation Units," *IEEE Access*, 2019. https://www.doi.org/10.1109/ACCESS.2019.2959036.

[13] S. C. Douglas and J. Yu, "Why RELU Units Sometimes Die: Analysis of Single-Unit Error Backpropagation in Neural Networks," *Conf. Rec. - Asilomar Conf. Signals, Syst. Comput.*, vol. 2018-October, no. 3, pp. 864–868, 2019. https://arxiv.org/pdf/1812.05981.

[14] J. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," *EMNLP-IJCNLP 2019 - 2019 Conf. Empir. Methods Nat. Lang. Process. 9th Int. Jt. Conf. Nat. Lang. Process. Proc. Conf.*, pp. 6382–6388, 2020. https://arxiv.org/pdf/1901.11196.

[15] Y. Xu and R. Goodacre, "On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning," *J. Anal. Test.*, 2018. https://www.doi.org/10.1007/s41664-018-0068-2.