IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 21 September 2023, accepted 16 October 2023, date of publication 23 October 2023, date of current version 1 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3326475

RESEARCH ARTICLE

Cyclical Learning Rate Optimization on Deep Learning Model for Brain Tumor Segmentation

AZIZ FAJAR[1,2], RIYANARTO SARNO[1], (Senior Member, IEEE), CHASTINE FATICHAH[1], (Member, IEEE), RAHADIAN INDARTO SUSILO[3], AND GUSTI PANGESTU[4]

[1]Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember, Surabaya 60111, Indonesia
[2]Department of Data Science Technology, Faculty of Advanced Technology and Multidiscipline, Universitas Airlangga, Surabaya 60115, Indonesia
[3]Department of Neurosurgery, Faculty of Medicine, Universitas Airlangga, Surabaya 60115, Indonesia
[4]School of Computer Science, Bina Nusantara University, Jakarta 11480, Indonesia

Corresponding author: Riyanarto Sarno (riyanarto@if.its.ac.id)

This work was supported in part by the Asian Development Bank under Higher Education for Technology and Innovation Asian Development Bank (HETI ADB) Project, in part by the Indonesian Ministry of Education and Culture under Penelitian Terapan Unggulan Perguruan Tinggi (PTUPT) Program, and in part by Institut Teknologi Sepuluh Nopember (ITS) under Project Scheme of the Publication Writing and Intellectual Property Rights (IPR) Incentive Program (PPHKI).

ABSTRACT In recent years, deep learning has found widespread applications in tasks such as segmentation and classification. Fine-tuning hyperparameters is crucial to improve performance, with the learning rate being a key parameter. Various methods, including adaptive learning rates, learning rate scheduling, and cyclical learning rates, have been used to optimize learning rates. Cyclical learning rates offer significant benefits with minimal computational cost, as seen in previous research. This study introduces a novel approach to tuning the cyclical learning rate, which incorporates the exponential moving average. These methods are applied to the BraTS 2021 dataset for segmentation tasks, resulting in superior performance compared to the previous approach. Our proposed method reduces the epochs required to reach convergence by 19 and 54 epochs for U-Net and Dense U-net, respectively. For Res U-net, the epoch needed to convergence is 10 epochs more. However, the proposed method produces lower loss values with 0.707, 0.657, and 0.665 compared to the previous method with 0.712, 0.685, and 0.725 for U-net, Res U-net, and Dense U-net, respectively.

INDEX TERMS 3D image data, deep learning, exponential moving average, information and communication technology, learning rate, scientific research.

I. INTRODUCTION

Research on medical image has recently taken the hotspot. Medical images are commonly found in the form of Digital Imaging and Communications in Medicine (DICOM). However, for brain medical images, the image may be in the form of Neuroimaging Informatics Technology Initiative (NIfTI) images. Many tasks involving medical image have been significantly performed in recent years [1], [2], [3], [4], [5]. Tasks such as segmentation [6], [7], [8], classification [9], [10], [11], [12], and reconstruction [13] have been performed for the subject using both private and public datasets.

Segmentation and classification tasks are done mostly using deep learning with different architectures and modifications [14], [15]. U-net is one of the most widely used architectures in medical image processing [16], [17], [18]. Thus, modifications on the U-net architecture emerged such as residual U-net (Res U-net) [19] and dense U-net [20]. One of the most important elements when building a deep learning model is the hyperparameters. The learning rate is considered the most important hyperparameter to adjust when training a deep learning model [21], [22], [23]. While setting the learning rate too low will slowly try to set the deep learning model to reach the most optimal result, setting the learning rate too high is not an option, since the result will diverge even after convergence.

The associate editor coordinating the review of this manuscript and approving it for publication was Vishal Srivastava.

There are multiple learning rate optimization methods, namely adaptive learning rates, time-based learning rate schedule, step-based learning rate schedule, and cyclical learning rate [24]. However, recent studies on learning rate optimization work on adaptive learning rate [25], [26]. Although the adaptive learning rate is effective, it is computationally expensive when processing large 3D image data. Moreover, the needs to optimize the initial learning rate using grid search are both time consuming and computationally expensive. Therefore, the alternative of adaptive learning rates is the Cyclical Learning Rates (CLR).

CLR are better compared to other learning rate tuning methods in terms of computational power and training results [21]. CLR uses lower computational power compared to the adaptive learning rate, yet still produces respectable training outcomes. Other learning rate optimization scheduling methods can vary the learning rate used while training similarly to CLR. However, they need scheduling techniques to determine when to reduce or increase the learning rate. In addition, several training sessions are needed to determine the schedule. This is not feasible while training data with high memory consumption and computational power. CLR reduces and increases the learning rate at set intervals and shows near-optimal results in the classification task in a previous study [21], [27].

Therefore, with the aforementioned benefit of the cyclical learning rate, this study proposes a method to improve CLR. The contributions of the study are the following:

1. This study introduces a methodology to establish the optimal range of initial learning rates for CLR by combining the learning rate test with the Exponential Moving Average (EMA). This approach obviates the necessity of iterative model training, such as grid searching, which is notorious for its time-intensive nature. Grid search can take several months to determine the optimal learning rate, since a single-model training iteration typically takes approximately one to two weeks.

2. This study also introduces a new method for the learning rate test. Using the learning rate we obtained from each training step of the epochs used for the learning rate test instead of the learning rate when each epoch ends.

3. Using the proposed method on the deep learning model for tumor segmentation using MICCAI Brain Tumor Segmentation (BraTS) 2021 data.

BraTS has been one of the most widely used public datasets with 1251 and 219 3D NIfTI images in the BraTS 2021 training and validation dataset, respectively [28], [29]. Tasks are divided into segmentation and classification with an online validation system. Thus, the results of the model can be uploaded and scored objectively. The 3D images in the segmentation dataset are brain imaging with tumors in 4 different modalities for each patient. The modalities are FLAIR, T1, T1 contrast-enhanced, and T2. Segmentation ground truths are provided in the training dataset, while in the validation dataset only the four brain modalities are provided.

II. PREVIOUS STUDIES

Previous research on learning rate optimization uses different approaches such as time-based learning rate schedule, step-based learning rate schedule, adaptive learning rates, and cyclical learning rate. The time-based learning rate schedule reduces the learning rate over time based on the set decay rate (d). Thus, it is a function with linear denominator [30] as shown in (1).

$$\eta_{n+1} = \frac{\eta_n}{1 + dn} \tag{1}$$

where n is the number of iterations and η_n denotes the current learning rate on the n iteration. Thus, using (1) the learning rate will be reduced over time.

The step-based learning rate schedule is based on the time-based learning rate schedule with some modifications. However, the step-based learning rate schedule uses an exponential function instead of a simple rational function. The step-based learning rate schedule can be calculated using (2).

$$\eta_n = \eta_0 d^{floor(\frac{1+n}{r})} \tag{2}$$

where n denotes the step, η_0 shows the value of the initialized learning rate, d is the decay rate, and r denoted the drop rate [31].

Adaptive learning rates are based on changing the learning rates in the optimization step. Adaptive learning rates such as Adaptive Gradient (Adagrad) change the learning rate by dividing the learning rate with the square of the gradient in each step. Thus, resulting in a higher denominator as training progresses [32].

Another method of adaptive learning rates is Root Mean Square Propagation (RMSProp). This adaptive learning rate divides the learning rate using the square root of the exponential moving average of the gradient square. This exponential moving average is also used in the most popular adaptive learning rate method, Adaptive Moment Estimation (Adam). Adam is based on the same update rule used in Adagrad and the use of the exponential moving average of RMSProp [33], [34].

Cyclical learning rates (CLR) aim to decrease and increase the learning rate over time. This method may have a short-term negative effect on the training process, shown by the increase in the loss value. However, in the long term, it is beneficial to make such a change in the learning rate as shown in previous studies [21], [27]. Cyclical learning rates may have different functional forms, such as triangular, parabolic, sinusoidal, and trapezoidal.

What is interesting about the cyclical learning rate is that we can set the maximum and minimum learning rates for the CLR. Therefore, optimizing the maximum and minimum bounds of the learning rate is crucial. Previous study [21] shows that using the learning rate range test may provide the model with the optimal learning rate range. This is done by plotting the accuracy of the classification task and the learning rate. However, the learning rate test can produce

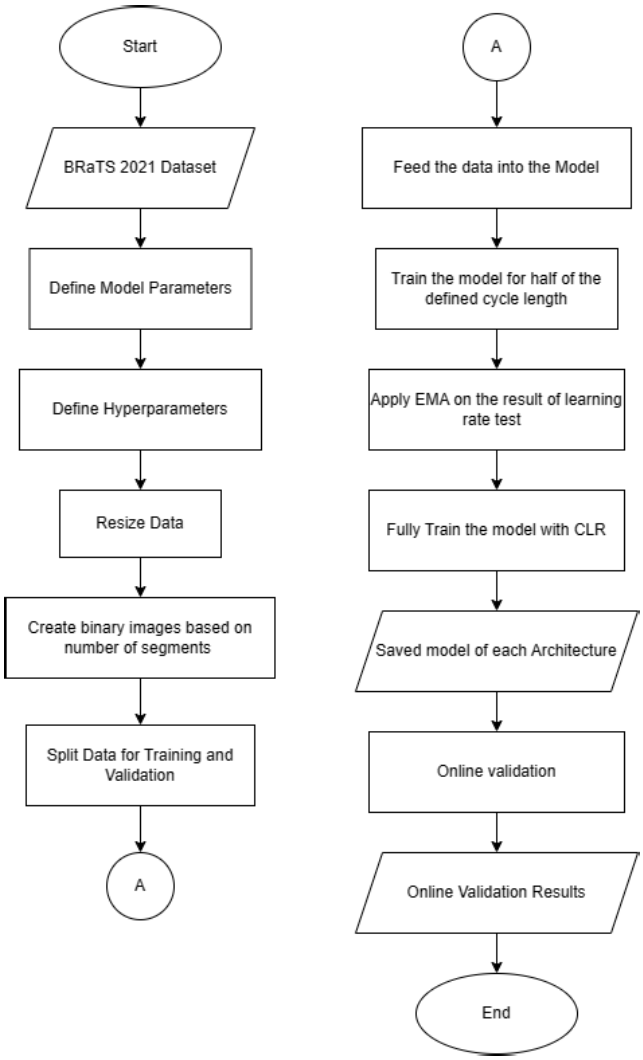

FIGURE 1. Flowchart of this research.

different graphs based on the datapoints used. These different ranges can result in different accuracy results. This study will examine how the learning rate range can be chosen.

The moving average and its variants have been used in several tasks, such as forecasting and trend analyzing. The simple moving average takes the average values of the given period [35]. Resulting in datapoints which shows the trend based on the time window used. A variant of a simple moving average is the Exponential Moving Average (EMA). The EMA has been used to calculate the trend of power consumption [36]. The difference between both methods is in the use of a time window for the simple moving average and a smoothing value for the EMA. The EMA is better for complex data as it responds to data trends more quickly.

III. METHODS

In this section we will explain our approach to segment the BraTS 2021 dataset using the proposed method shown in Figure 1.

A. PROPOSED METHOD

Our proposed method consists of the use of an improved learning rate test and trends of the learning rate tests to determine the lower and upper bounds of the CLR.

1) LEARNING RATE TEST

The learning rate test requires a training session with a number of epochs equal to half the cycle length of the CLR. To determine the cycle length, we used the method of the previous study [21]. The starting learning rate range used is 0.0001 to 0.01. Therefore, the learning rate test result has a wide range of learning rate that can be used. The difference between our proposed method and the previous is the number of datapoints used. In this study, we take the learning rate of each training step within each epoch. Producing graphs with more datapoints. The result of the learning rate test is then used to determine the optimal learning rate for the current architecture with the given dataset. The learning rate tests using our proposed method produce graphs as shown in Figure 2.

Figure 2 shows the results of the learning rate test for 3 different architectures. Figure 2 shows the learning rate test of (a) U-Net, (b) Res U-net, and (c) Dense U-Net. The learning rate tests were performed in 10 epochs while we record the learning rate of each training step. From 2 we can learn that each architecture has a different optimal learning rate. Based on a previous study [21], the optimal learning rate for the U-Net architecture based on the learning rate test in Figure 2(a) would be 0.003 to 0.004 since the loss value starts to decrease when the learning rate is 0.003, and starts to increase when the learning rate is 0.004. However, using a cyclical learning rate with small difference between the lower and upper bounds might keep the gradient stuck at a local minimum. Thus, instead of using the learning rate range where the loss value starts to decrease for the lower bounds and the learning rate where the loss starts to increase for the upper bounds for each architecture, we propose a different approach.

2) EXPONENTIAL MOVING AVERAGE

In this study, we used the trends of the learning rate to determine the lower and upper bounds of the CLR. To calculate the trends, we used the Exponential Moving Average (EMA) as shown in (3).

$$y_n = (1 - \alpha_n)y_{n-1} + \alpha_n x \tag{3}$$

where:

a_n = learning rate at time n

x = smoothing value in range of $0 < x < 1$

y_n = EMA at time n

EMA calculates the short-term trends of the data. Therefore, it is suitable for the learning rate test since the use of a method that calculates long-term trends would make the learning rate range test too long and require more

FIGURE 2. Learning rate test of (a) U-Net; (b) Res U-Net; and (c) Dense U-Net.

computational power, which is contrary to the objective of this study. With the trends of each set of data with the given architecture, we select the learning rate range that decreases the loss value the most throughout the training sessions. Thus, it gives information on the most optimal learning rate range for the whole dataset when it is trained using the given architecture.

IV. RESULTS AND DISCUSSION

In this section we will discuss the results of the proposed method with the methodology mentioned in the methods section.

A. DATA PREPROCESSING

The data we used in this study is the BraTS 2021 dataset with 1251 training data for segmentation tasks. We trained the whole 3D image data of size 240 × 240 × 240. Since BraTS data included in the dataset have 240 × 240 × 155, only 155 voxels in the last dimension, we add zero padding to the images for the last dimension. We divided the ground truth into 4 segments. The number of segments is based on the number of unique voxels in the ground truth. Each ground truth provided by BraTS only has voxels values of 0, 1, 2, and 4. These segments are comprised of background (BG) with voxels value of 0, tumor core (TC) with voxels value of 1, peritumoral edema (ED) with voxels value of 2, and GD-enhancing tumor (ET) with voxels value of 4. This is done to increase the precision of the segmentation while training the model. Dividing the ground truth into 4 segments can increase the precision of the segmentation results, since each segment now consists of only 0 and 1 valued voxels. Each architecture used ReLU for the activation function of each layer and softmax for the activation function of the last layer to produce 4 images as segments for the segmentation result. Figure 3 shows the divided ground truth images.

Figure 3 shows the divided ground truth image in four segments. These four segments will be compared with the segments produced by each deep learning architecture, and the result will be calculated using the Dice Similarity Coefficient (DSC).

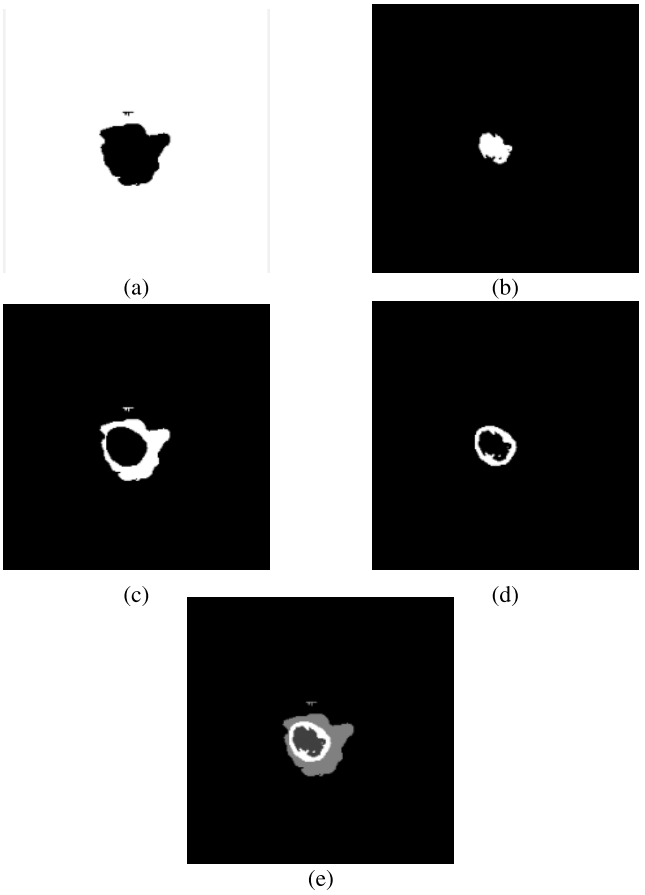

FIGURE 3. BraTS 2021 ground truth images divided into 4 parts, (a), (b), (c), (d), and the original ground truth shown in (e).

B. TRAINING SETUP

The training processes were carried out using a computer utilizing a NVIDIA Quadro 8000 with 48 GB dedicated memory for the GPU, 64 GB RAM and Intel I9 for the processor. For the software, we are using Python 3.7.0 and Tensorflow 2.7.0 with a custom training loop.

In this study, we used the triangular learning rate cycle as the functional form for the CLR. The CLR step size is set to 10 epochs with a cycle length of 20. The CLR step size determines the number of epochs in which the learning rate

FIGURE 4. Learning rate test result of (a) U-Net, (b) Res U-net, and (c) Dense U-Net architecture.

test is performed. We decayed the learning rate after each cycle, as decayed cyclic learning rates show the best result over the nondecayed cyclic learning rates in the previous studies. The architecture used in this study is the U-Net architecture and its modifications such as Res U-Net and Dense U-Net with DSC as the loss function. However, the memory required to train 3D image data with the mentioned size is huge. Thus, to avoid memory exhaustion, a custom training loop was created. The custom training loop loads a batch of 4 images and divides it into 75:25 for training and validation data for each training step in an epoch. Then, another 4 images will be retrieved for the next training step until all the data are trained in an epoch. We proposed an improved learning rate test to optimize the learning rate used for the model. We also set an early stop to the training if the validation loss does not improve after 20 epochs.

C. LOSS FUNCTION

For this study, the loss function used is the Dice Similarity Coefficient (DSC). However, since our deep learning networks produce 4 segments, the DSC of each segment is calculated, and the total value of the DSC from all segments is used as the loss value of the epoch. Equation (4) shows how the DSC is calculated for each segment, while (5) shows how the total DSC is calculated at the end of each epoch.

$$DSC = \frac{2 \times |X \cap Y|}{|X| + |Y|} \tag{4}$$

$$DSC_{total} = 4 - (d_{bg} + d_{TC} + d_{ED} + d_{ET}) \tag{5}$$

where, DSC denotes the DSC of a segment, while d_{total} denotes the total loss value. Based on DSC_{total}, the loss value of each training step will be summed and divided by the number of training steps to obtain the total loss value of the epoch. Based on (5), the maximum loss value is 4 and the minimum is 0.

D. LEARNING RATE TEST RESULTS

Using our proposed method, we created the learning rate test graphs and calculated the EMA of the learning rate test for each architecture. The CLR will use the learning rate in the range we set for a cycle, then decay the upper bounds for the next cycle. However, for the learning rate test, we only train the model for half of the cycle. The result of the learning rate test used in our proposed method is shown in Figure 4.

Figure 4 shows the learning rate test for different architecture results. The blue and orange lines indicate the effect of the learning rate on the loss value and the loss value EMA of the current learning rate test, respectively. The results of the learning rate test and the loss value EMA of (a) U-Net, (b) Res U-net, and (c) Dense U-Net show different start when the loss value starts to decrease and increase. Based on the results shown by the orange line in Figure 3 (a), the learning rate range that is the most optimal is between 0.0035 where the loss value starts to decrease and 0.0055 where the loss value starts to increase. Figure 3 (b) shows 0.004 to 0.005 as the most optimal. Figure 3 (c) shows that 0.0045 to 0.006 is the most optimal. Thus, we can decide the learning rate range of each architecture for the given dataset.

In Figure 4 (a), the orange line in the learning rate range of 0.0035 to 0.0055 shows that the trend of the loss value is still decreasing or constant. The loss value is the lowest when the learning rate is in the range of 0.004 and 0.005, then starts to increase at the learning rate of 0.0055. On the other hand, the previous study [21] performed the learning rate test to determine the lower and upper bounds differently. Instead of using the learning rate used in each training step, the previous study takes the learning rate used in each epoch ends. Figure 5 shows how the previous research performed the learning rate test.

Based on Figure 5, each architecture has a different optimal learning rate range. In Figure 5 (a), the upper bound will be set at 0.0055 as the loss value begins to increase at the given learning rate, while the lower bound is set at about 0.0045. Figure 5 (b) shows the most optimal learning rate range of 0.0065 to 0.0075, and Figure 5 (c) shows that the best range is at 0.0095 to 0.01.

E. TRAINING RESULT

The time needed for each epoch of training of each model is different. For each training step within each epoch, U-Net needs about 14.20 seconds to 14.30 seconds, Res U-net needs 14.40 to 14.50 seconds, and Dense U-Net needs 15.00 seconds on average. Thus, U-Net is the fastest for the training time on each epoch. Figure 6 shows the training

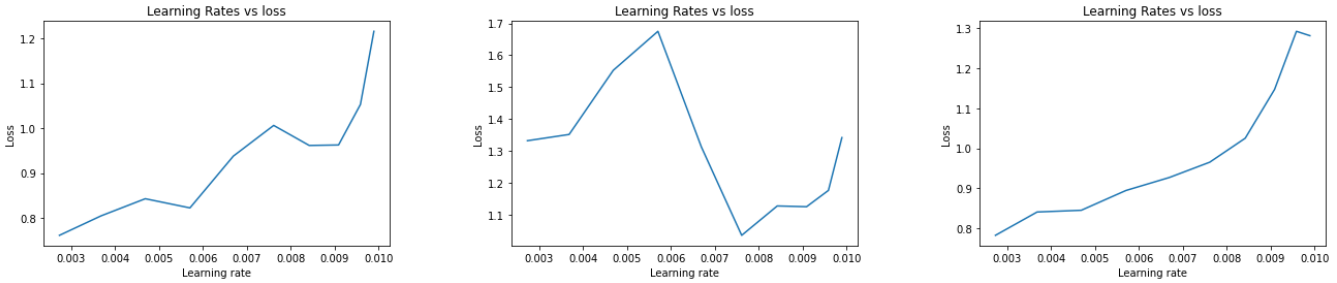

FIGURE 5. Learning rate tests of (a) U-Net, (b) Res U-Net, and (c) Dense U-Net using method used in previous study.

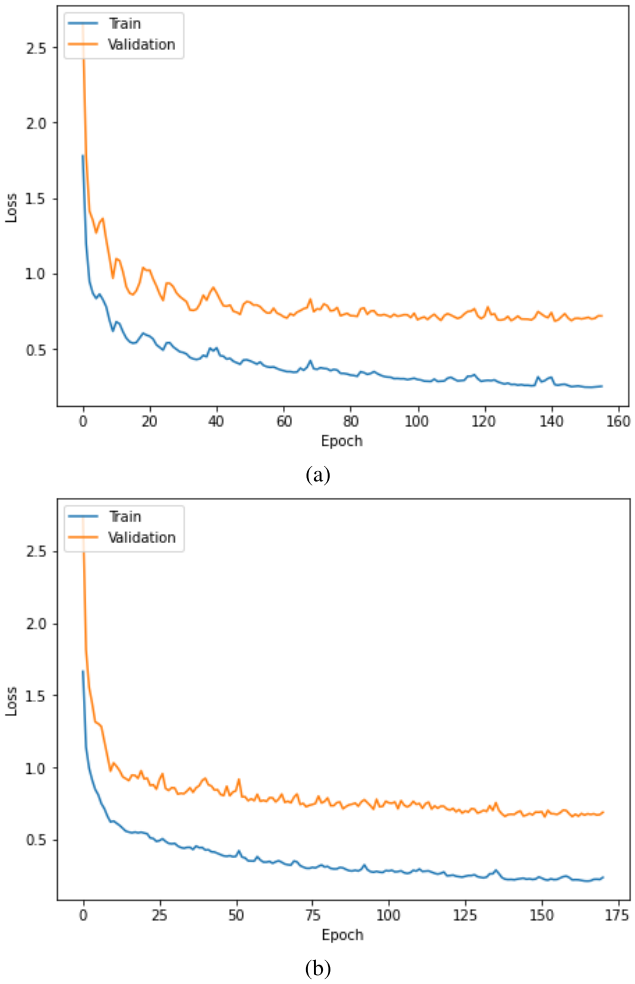

FIGURE 6. Loss graph of Res U-Net with (a) tuned using method on previous research, and (b) tuned using proposed method.

loss result of the Res U-net architecture using (a) the method used in the previous study and (b) tuned using the proposed method. The final model of both methods starts to converge in about 120 epochs. However, the proposed method produces a lower loss value on the same epoch than the previous method and improved after several epochs. Thus, indicating that the proposed method is better and converges faster. The final training loss for each architecture is shown in Table 1.

Table 1 shows that our proposed method is better during model training. The loss value is in the range of 0 to 4. Both the loss value and the DSC show that the proposed method produces better results in the validation results during the training process. The training time required for each model to converge is shown in Table 2.

Table 2 shows that our proposed method can reach convergence faster than the previous method except for Res U-Net. All architectures that employ the proposed method attain the optimal validation result of the previous method faster and consistently exhibit superior performance, resulting in improved validation results. Using the previous method, Res U-Net produced the best loss value in the validation set generated from the training dataset in 142 epochs. However, by using our proposed approach, the architecture achieves almost identical results, with a loss value of 0.6838005 over 125 epochs. Thereafter, it continues to improve and displays the best results at 152 epochs. The Dense U-Net result shows the most significant results. Since Dense U-Net requires higher computational power due to the dense block used, the training time of this architecture is generally slower than two other architectures. Using our proposed method, the network training time can be significantly faster. However, our method produces better segmentation results and requires a shorter training time based on the results on 2. Figure 7 shows the segmentation result of the previous method and the proposed method for the Res U-Net architecture.

Based on Figure 7, the result of the segmentation of the proposed method shown in Figure 7 (b) and the result of the previous method in Figure 7 (c) compared to the ground truth shown in Figure 7 (a) shows only few differences. Figure 8 shows the detailed results of the segmentation difference.

Figure 8 shows how the results of the segmentation overlapped each other. The orange line surrounds the area in which all images in 7 show the most obvious difference of the TC and ET segment, while red and turquoise show the difference in the ED segment. Based on Figure 8, the segmentation result produced by our proposed method is more similar to ground truth compared to the previous study in areas within the orange and turquoise line. However, the previous study showed a better segmentation result in the area surrounded by the red line.

TABLE 1. Validation results during training process.

Architecture	Loss Value		DSC	
	Previous Method	Proposed Method	Previous Method	Proposed Method
U-Net	0.712	0.707	82.2%	82.325%
Res U-Net	0.685	0.657	82.875%	83.575%
Dense U-Net	0.725	0.665	81.875%	83.375%

TABLE 2. Time performance result of the methods used.

Architecture	Time per Training Step		Epoch Needed to Convergence	
	Previous Method	Proposed Method	Previous Method	Proposed Method
U-Net	15.15s	14.78	140	121
Res U-Net	15.27s	14.66	142	152
Dense U-Net	35s	34s	224	170

TABLE 3. Online validation results.

Architecture		Dice			Sensitivity			Specificity			Precision		
		ET	TC	WT	ET	TC	WT	ET	TC	WT	ET	TC	WT
U-Net	Previous Study	0.6865	0.7323	0.8626	0.6880	0.7350	0.8576	0.9984	0.9995	0.9995	0.7902	0.8921	0.9220
	Proposed method	0.7223	0.7208	0.8702	0.7324	0.7143	0.8662	0.9990	0.9987	0.9995	0.8010	0.8890	0.9234
Res U-Net	Previous Study	0.6958	0.7342	0.8527	0.6965	0.7002	0.8292	0.9996	0.9996	0.9990	0.7833	0.8839	0.9235
	Proposed method	0.7179	0.7594	0.8357	0.7263	0.7391	0.8093	0.9988	0.9989	0.9982	0.8004	0.8871	0.9176
Dense U-Net	Previous Study	0.6877	0.7213	0.8601	0.7020	0.7254	0.8545	0.9988	0.9990	0.9996	0.7701	0.8800	0.9147
	Proposed method	0.7033	0.7333	0.8340	0.7330	0.7237	0.8443	0.9994	0.9990	0.9990	0.8010	0.8911	0.9186

(a) (b) (c)

FIGURE 7. Images of (a) ground truth, (b) segmentation result using proposed method, and (c) segmentation result using method in previous study using Res U-Net architecture.

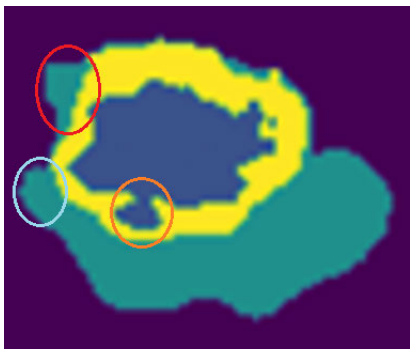

FIGURE 8. Overlapping parts of a segmentation result.

F. ONLINE VALIDATION RESULT

BraTS 2021 has a separate dataset that contains 219 sets of data for the validation dataset. This dataset does not provide the ground truth image. Instead, the segmentation result is sent to a system to validate the model. The segmentation result is divided into Enhancing Tumor (ET), Tumor Core (TC), and Whole Tumor (WT). Table 3 shows the result of the online validation system.

Based on Table 3, the proposed method has achieved better results in several metrics. DSC of ET and TC of every architecture that is using the proposed method is superior to the previous method, except for the DSC of TC with the U-Net architecture. However, the DSC of WT for the U-Net architecture is best when the proposed method is used. The sensitivity of the segmentation result tends to be similar to that of the DSC of each segment. Therefore, resulting in higher sensitivity compared to the other method when the DSC of the given method is higher. Almost all architectures produce high specificity. With the highest value of 0.9996 and the lowest value of 0.9982. However, for the precision metric,

our proposed method increases the result in almost every architecture, except for the precision of WT with Res U-net architecture. Based on the training result, the image produced by architectures when using our proposed method might have a lower DSC in the ED area but performed better when segmenting the ET area. Therefore, the lower the DSC value when segmenting the WT, since ED makes up the largest part of the WT.

V. CONCLUSION

We proposed a new method to tune the learning rate of deep learning models using CLR and EMA. EMA is used to determine the trend of a set of data. Using the learning rate trend toward the loss value, we were able to produce a better result than in the previous study. Our proposed method reduces the epochs required to reach convergence by 19 and 54 epochs for U-Net and Dense U-net, respectively. For Res U-net, the epoch needed to convergence is 10 epochs more. However, the proposed method produces lower loss values with 0.707, 0.657, and 0.665 compared to the previous method with 0.712, 0.685, and 0.725 for U-net, Res U-net, and Dense U-net, respectively.

We used the method in a previous study to determine the CLR cycle length. However, in our opinion the cycle length set can be more suited if it was determined by the architecture used. In our research, Dense U-net, as the most computationally expensive and slowest to train, requires a longer training time. Therefore, the learning rate is too low even before reaching near the end of the training, resulting in a longer training time. Hence, future work of this research can tackle the problem mentioned in optimizing the cycle length based on how computationally expensive the architecture is.

REFERENCES

[1] S. F. Qadri, H. Lin, L. Shen, M. Ahmad, S. Qadri, S. Khan, M. Khan, S. S. Zareen, M. A. Akbar, M. B. Bin Heyat, and S. Qamar, "CT-based automatic spine segmentation using patch-based deep learning," *Int. J. Intell. Syst.*, vol. 2023, pp. 1–14, Mar. 2023, doi: 10.1155/2023/2345835.

[2] Y. Zhuang, S. Chen, N. Jiang, and H. Hu, "An effective WSSENet-based similarity retrieval method of large lung CT image databases," *KSII Trans. Internet Inf. Syst.*, vol. 16, no. 7, pp. 2359–2376, 2022, doi: 10.3837/tiis.2022.07.013.

[3] S. Lu, B. Yang, Y. Xiao, S. Liu, M. Liu, L. Yin, and W. Zheng, "Iterative reconstruction of low-dose CT based on differential sparse," *Biomed. Signal Process. Control*, vol. 79, Jan. 2023, Art. no. 104204, doi: 10.1016/j.bspc.2022.104204.

[4] M. Liu, X. Zhang, B. Yang, Z. Yin, S. Liu, L. Yin, and W. Zheng, "Three-dimensional modeling of heart soft tissue motion," *Appl. Sci.*, vol. 13, no. 4, p. 2493, Feb. 2023, doi: 10.3390/app13042493.

[5] S. Lu, S. Liu, P. Hou, B. Yang, M. Liu, L. Yin, and W. Zheng, "Soft tissue feature tracking based on deep matching network," *Comput. Model. Eng. Sci.*, vol. 136, no. 1, pp. 363–379, 2023, doi: 10.32604/cmes.2023.025217.

[6] M. Ahmad, S. F. Qadri, M. U. Ashraf, K. Subhi, S. Khan, S. S. Zareen, and S. Qadri, "Efficient liver segmentation from computed tomography images using deep learning," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–12, May 2022, doi: 10.1155/2022/2665283.

[7] A. S. Akbar, C. Fatichah, and N. Suciati, "Single level UNet3D with multipath residual attention block for brain tumor segmentation," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 6, pp. 3247–3258, Jun. 2022, doi: 10.1016/j.jksuci.2022.03.022.

[8] U. Salamah, R. Sarno, A. Z. Arifin, A. S. Nugroho, I. E. Rozi, and P. B. S. Asih, "A robust segmentation for malaria parasite detection of thick blood smear microscopic images," *Int. J. Adv. Sci., Eng. Inf. Technol.*, vol. 9, no. 4, pp. 1450–1459, 2019, doi: 10.18517/ijaseit.9.4.4843.

[9] L. Sun, M. Zhang, B. Wang, and P. Tiwari, "Few-shot class-incremental learning for medical time series classification," *IEEE J. Biomed. Health Informat.*, early access, Feb. 22, 2023, doi: 10.1109/JBHI.2023.3247861.

[10] A.-C. Phan, T.-M.-N. Nguyen, and T.-C. Phan, "Detection and classification of brain hemorrhage based on Hounsfield values and convolution neural network technique," in *Proc. IEEE-RIVF Int. Conf. Comput. Commun. Technol. (RIVF)*, Mar. 2019, pp. 1–7, doi: 10.1109/RIVF.2019.8713733.

[11] Z. Jia and D. Chen, "Brain tumor identification and classification of MRI images using deep learning techniques," *IEEE Access*, early access, Aug. 13, 2020, doi: 10.1109/access.2020.3016319.

[12] A. Rehman, M. A. Khan, T. Saba, Z. Mehmood, U. Tariq, and N. Ayesha, "Microscopic brain tumor detection and classification using 3D CNN and feature selection architecture," *Microsc. Res. Technique*, vol. 84, no. 1, pp. 133–149, Jan. 2021, doi: 10.1002/jemt.23597.

[13] A. Fajar, R. Sarno, C. Fatichah, and A. Fahmi, "Reconstructing and resizing 3D images from DICOM files," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 6, pp. 3517–3526, Jun. 2022, doi: 10.1016/j.jksuci.2020.12.004.

[14] G. Pangestu, H. L. Hendric Spits Warnars, B. Soewito, and F. L. Gaol, "The use of deep and machine learning for face expression recognition: A literature review," in *Proc. Int. Conf. Inf. Manag. Technol. (ICIMTech)*, Aug. 2022, pp. 201–206, doi: 10.1109/ICIMTech55957.2022.9915257.

[15] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, p. 53, Mar. 2021, doi: 10.1186/s40537-021-00444-8.

[16] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI*. Cham, Switzerland: Springer, 2016, pp. 424–432, doi: 10.1007/978-3-319-46723-8_49.

[17] H.-J. Bae, H. Hyun, Y. Byeon, K. Shin, Y. Cho, Y. J. Song, S. Yi, S.-U. Kuh, J. S. Yeom, and N. Kim, "Fully automated 3D segmentation and separation of multiple cervical vertebrae in CT images using a 2D convolutional neural network," *Comput. Methods Programs Biomed.*, vol. 184, Feb. 2020, Art. no. 105119, doi: 10.1016/j.cmpb.2019.105119.

[18] B. Wu, Y. Fang, and X. Lai, "Left ventricle automatic segmentation in cardiac MRI using a combined CNN and U-Net approach," *Computerized Med. Imag. Graph.*, vol. 82, Jun. 2020, Art. no. 101719, doi: 10.1016/j.compmedimag.2020.101719.

[19] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual U-Net," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 5, pp. 749–753, May 2018.

[20] M. Kolarík, R. Burget, V. Uher, K. Říha, and M. Dutta, "Optimized high resolution 3D dense-U-Net network for brain and spine segmentation," *Appl. Sci.*, vol. 9, no. 3, p. 404, Jan. 2019, doi: 10.3390/app9030404.

[21] L. N. Smith, "Cyclical learning rates for training neural networks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 464–472, doi: 10.1109/WACV.2017.58.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1–9, doi: 10.1145/3065386.

[23] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, doi: 10.1007/978-3-642-35289-8_13.

[24] S. Anter, H. K. Hussein, M. H. Ali, and F. F. Mohamed, "Cyclic self-organizing map for object recognition," *Inf. Sci. Lett.*, vol. 15, no. 5, pp. 1775–1788, 2023, doi: 10.18576/isl/120523.

[25] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," in *Proc. ICLR*, 2019, pp. 1–14.

[26] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*.

[27] J. Li and X. Yang, "A cyclical learning rate method in deep learning training," in *Proc. Int. Conf. Comput., Inf. Telecommun. Syst. (CITS)*, Oct. 2020, pp. 1–5, doi: 10.1109/CITS49457.2020.9232482.

[28] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, and L. Lanczi, "The multimodal brain tumor image segmentation benchmark (BRATS)," *IEEE Trans. Med. Imag.*, vol. 34, no. 10, pp. 1993–2024, Oct. 2015, doi: 10.1109/TMI.2014.2377694.

[29] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. S. Kirby, J. B. Freymann, K. Farahani, and C. Davatzikos, "Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features," *Sci. Data*, vol. 4, no. 1, Sep. 2017, Art. no. 170117, doi: 10.1038/sdata.2017.117.

[30] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, vol. 87. Boston, MA, USA: Springer, 2004, doi: 10.1007/978-1-4419-8853-9.
[31] R. Ge, S. M. Kakade, R. Kidambi, and P. Netrapalli, "The step decay schedule: A near optimal, geometrically decaying learning rate procedure for least squares," in *Proc. Adv. Neural Inf. Process. Syst.*, Apr. 2019, pp. 1–12.
[32] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," in *Proc. 23rd Conf. Learn. Theory*, vol. 12, 2010, pp. 257–269.
[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," Dec. 2014, *arXiv:1412.6980*.
[34] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," 2019, *arXiv:1904.09237*.
[35] M. H. P. Swari, I. P. S. Handika, and I. K. S. Satwika, "Comparison of simple moving average, single and modified single exponential smoothing," in *Proc. IEEE 7th Inf. Technol. Int. Seminar (ITIS)*, Oct. 2021, pp. 1–5, doi: 10.1109/ITIS53497.2021.9791516.
[36] Y. Zhou and L. Yang, "Control strategy to smooth wind power fluctuations of PMSG wind turbine based on the secondary exponential moving average method," in *Proc. IEEE PES Asia–Pacific Power Energy Eng. Conf. (APPEEC)*, Dec. 2019, pp. 1–5, doi: 10.1109/APPEEC45492.2019.8994429.

AZIZ FAJAR was born in Malang, Jawa Timur, Indonesia. He is currently pursuing the Ph.D. degree with Institut Teknologi Sepuluh Nopember Surabaya, focusing on computer vision, especially medical image processing. He has been working on projects with neurosurgeon to process MRI, CT scan, and MRA images to help neurosurgeons treats Parkinson diseases and brain tumor, since 2019. Up to now, he has several publications on the subjects in the form of journals and conferences using both machine learning and CNN methods. He has been with the Department of Data Science Technology, Airlangga University, as a Lecturer, since 2023.

RIYANARTO SARNO (Senior Member, IEEE) received the Ph.D. degree, in 1992. He is currently a Professor with the Informatics Department, Institut Teknologi Sepuluh Nopember (ITS). His research interests include machine learning, deep learning, generative AI, the Internet of Things, and knowledge engineering, which have been implemented in electronic noses, medical equipment, and health care systems in recent years. He wrote more than eight books and more than 573 scientific articles, which led him to the top 2% world ranking scientist, in 2020 and 2023, by Standford University.

CHASTINE FATICHAH (Member, IEEE) was born in Pasuruan, in December 1975. She received the bachelor's degree in informatics, in 2000, the master's degree in computer science, in 2008, and the Ph.D. degree in computational intelligence, in 2012. Currently, she is a Professor with the Department of Informatics, Institut Teknologi Sepuluh Nopember (ITS). She also got some awards in her name, in 2014, such as the Young Researcher Award from Advanced Computational Intelligence and Intelligent Informatics (JACIII) by Fuji Press in Tokyo, Japan.

RAHADIAN INDARTO SUSILO was born in Madiun, in December 1977. He received the bachelor's degree in medical, in 2002, and the Ph.D. degree from Airlangga University, in 2019. Then, he continued his study to pursue Neurosurgery Specialist with Airlangga University, after he was graduated. He works in three hospitals in Surabaya, Indonesia, namely, Dr. Soetomo General Hospital, Premier Hospital, and Mitra Keluarga Hospital, as a Neurosurgeon Consultant.

GUSTI PANGESTU received the bachelor's (S.Kom.) degree from UIN Maulana Malik Ibrahim, Indonesia, and the master's (M.Kom.) degree from Brawijaya University, Indonesia. He is currently pursuing the Ph.D. degree with Bina Nusantara University, Indonesia. He is also a teacher and a researcher. He conducts research in the field of artificial intelligence (AI) concerning social interaction, computer vision, and mobile technology. In addition to his university teaching role, he is also involved as a student advisor for research and competitions.