CP_SDUNet: road extraction using SDUNet and centerline preserving dice loss

Bayu Satria Persada, Muhammad Rifqi Priyo Susanto, Laksmita Rahadianti, Aniati Murni Arymurthy
Faculty of Computer Science, Universitas Indonesia, Jakarta, Indonesia

Article Info

Article history:

Received Nov 6, 2024
Revised Jan 31, 2025
Accepted Mar 5, 2025

Keywords:

Connectivity preserving
CP_clDice
CP_SDUNet
Dice loss
Road extraction
SDUNet

ABSTRACT

Existing automatic road map extraction approaches on remote sensing images often fail because they cannot understand the spatial context of an image. Mainly because they could not learn the spatial context of the image and only knew the structure or texture of the image. These approaches only focus on regional accuracy instead of connectivity. Therefore, most approaches produce discontinuous outputs caused by buildings, shadows, and similarity to rivers. This study addresses the challenge of automatic road extraction, focusing on enhancing road connectivity and segmentation accuracy by proposing a network-based road extraction that uses a spatial intensifier module (DULR) and densely connected U-Net architecture (SDUNet) with a connectivity-preserving loss function (CP_clDice) called CP_SDUNet. This study analyzes the CP_clDice loss function for the road extraction task compared to the BCE Loss function to train the SDUNet model. The result shows that CP_SDUNet, performs best using an image size of 128×128 pixels and trained with the whole dataset with a combination of 20% clDice and 80% dice loss. The proposed method obtains a clDice score of 0.85 and an Interest over Union (IoU) score of 0.65 for the testing data. These findings demonstrate the potential of CP_SDUNet for reliable road extraction.

Corresponding Author:

Bayu Satria Persada
Faculty of Computer Science, Universitas Indonesia
South Jakarta, Jakarta, Java, Indonesia
Email: bayusatriapersada@gmail.com

1. INTRODUCTION

Automatic road map generation or road extraction from images benefits many applications, such as autonomous vehicles, urban planning, road network planning, and land investigation. Based on the data used, road extraction tasks can be divided into three types, i.e., road extraction using a 2D dataset, a 3D dataset, or both types simultaneously. Road extraction using a 2D dataset can be done using optical images [1], [2] or SAR images [3], [4], which each have their characteristics. This paper focuses on using 2D datasets, specifically optical images, to perform automatic road extraction.

The main problem in road extraction from satellite images is to label the road as foreground and other things as background, creating a binary segmentation problem. Compared with other image segmentation problems, many factors affect road extraction. The first is that roads are thin and long, spanning a large area even though their area (pixel count) is small. Next, the geometric features of roads are very similar to rivers and railways. These problems make it tricky even for professionals to differentiate between them. Additionally, roads can overlap in a complex intersection, and finally, in satellite images, roads can be obscured by trees, their shadows, and buildings' shadows.

In early studies, researchers designed hand-crafted features for road extraction [5]–[9]. This conventional technique uses texture structure, super-pixel partitioning [10], region-growing algorithm [11] and other methods for road extraction. However, satellite imagery has grown massively in recent years and produced very high-resolution images. These images bring new challenges for road extraction due to the complex structure of the road and diverse background distribution. With these new challenges, conventional methods start to underperform. In recent years, the development of neural networks such as convolutional neural networks (CNN) has improved in lots of semantic segmentation tasks, which includes road extraction [12]–[15]. Recently, the U-Net [16] architecture has been used extensively for segmentation tasks. Although CNN-based methods and U-Net are ubiquitous in segmentation tasks, they are often inadequate for remote sensing images. To address the issue, Yang *et al.* [17] designed a framework based on spatial-enhanced and densely connected U-Net to address the problem.

Additionally, the training strategies can also be modified to optimize models for extraction tasks. The model's loss function plays an important role as it guides the model in achieving the best result based on the difference between the predicted value and its ground truth. For a task that uses segmentation, in this case, road extraction, choosing the loss function is important. Several studies have proposed alternate loss functions that could increase the outcome of road extraction [18]. One of them uses structural similarity as a loss function, which is a joint loss of cross entropy and Dice loss called cross-entropy-dice-loss (CEDL) [19] function. However, none of them maintain the road connectivity. For that reason, Abdollahi *et al.* [20] proposed using CP_clDice. This new measure compares the intersection of pixels and their morphological skeleton to preserve the road's connectivity and obtain a better performance of the road extraction model.

In this paper, we proposed a novel approach to automatic road extraction. We introduce using spatial-enhanced and densely connected U-Net to capture spatial context from an image called CP_SDUNet. CP_SDUNet can produce a generally better accuracy on road segmentation using CP_clDice as its loss function to ensure road connectivity is preserved, which uses a spatial intensifier (DULR module) and densely connected U-Net with connectivity preserving loss function. By enhancing spatial context capture and explicitly maintaining road connectivity, this method aims to overcome limitations in existing techniques. Our work improves segmentation accuracy and ensures practical usability for applications that demand reliable road network extraction, such as urban infrastructure planning and autonomous vehicle systems.

2. METHODS

This work proposes a new road extraction method called connectivity preserving spatial enhanced and densely U-Net (CP_SDUNet). The proposed technique is a modification of the original deep learning model SDUNet [17], which is based on U-Net architecture but with better performance by enhancing spatial context. Additionally, it consists of a connectivity-aware similarity measure based on the intersection of the skeleton and its mask (CP_clDice) [20] to preserve connectivity of the road that has been extracted.

2.1. SDUNet

SDUNet [17] is a deep network architecture that is based on U-Net [16] and ResNet [21]. SDUNet consists of three major parts: decoder, bottleneck, and decoder. The encoder is filled with convolutional block and ReLU Activation, which later will be downsampled using Maxpooling, while the decoder is up-sampled with transpose convolution. The prominent feature of SDUNet is the use of a skip connection, which was inspired by ResNet [21]. Skip connection allows the sending of information without the need to go through layers of convolution. Hence, prevents vanishing gradient problems.

SDUNet is also inspired by DenseNet where each layer is connected to the other so that it is able to improve feature flows. The encoder consists of 4 dense blocks, the bottleneck consists of 1 dense block, and lastly, the decoder consists of 4 dense blocks [17].

2.1.1. Dense block

SDUNet is a series of dense blocks that mainly consist of multiple layers which extract information that is shaped like the letter U and are connected to each other, which was inspired by DenseNet architecture [22]. The blocks are connected to preserve the nature of forward feeding, where each layer will add additional information from all preceding layers and pass its feature map to subsequent layers. First, the input tensor passes through a batch normalization layer to re-scale and standardize its value. Then, it is activated to a rectified linear unit activation function, followed by a 3×3 convolution layer. SDUNet uses a modified version of DenseNet, where after going through a 3×3 convolution layer, the tensor moves to a dropout layer to drop some neurons randomly.

Figure 1 illustrates dense blocks used in SDUNet. The purpose of using dense blocks was to enhance information flow between layers. Consequently, the l^{th} layer receives the feature maps of all preceding layers, $x_0, \ldots, x_{l-1}$ as input. The formula is represented in (1).

$$x_l = H_t([x_0, \ldots, x_{l-1}]) \tag{1}$$

where $[x_0, \ldots, x_{l-1}]$ represent the concatenation of the feature maps generated from all layers beforehand and $H_t(.)$ refers to a nonlinear mapping function.

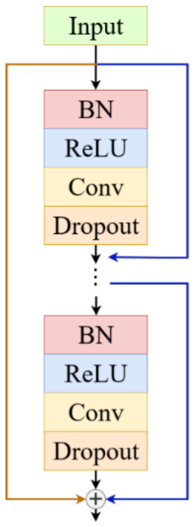


Figure 1. Dense block in SDUNet architecture

2.1.2. Transition layer

The transition layer consists of two parts: a down-transition layer and an up-transition layer. The down-transition layer is applied to downsample the feature map, which the dense block could not do. So, the transition layer consists of batch normalization to standardize, followed by a 1×1 convolution layer and a 2×2 pooling layer to downsampling. Conversely, the up-transition layer will perform the up-sampling using the same approach as the down-transition layer, but it uses a transposed convolution instead of a 2×2 pooling layer.

2.1.3. Down up left right convolution module (DULR)

SDUNet also proposed another way to spatially enhance the spatial context feature by doing what is called a DULR module [23]. Figure 2 illustrates the details of the DULR module architecture, which consists of up convolution, down convolution, left convolution, and right convolution. As shown in the figure, if we take down convolution and left convolution as an example, the left convolution will reduce the feature map from a size of C×W×H to C×1×H so that it will convolve along those H. Meanwhile, Down convolution will reduce a C×W×H feature map to become C×W×1 so that it convolves for the W tensor only. It will convolve every C×W×1 for every slice until it becomes the H again so that the feature map after it gets convoluted is back to C×W×H. Every convolution will be activated using a nonlinear function that is a rectified linear unit (ReLU). Similarly, the other convolution has the same function but goes in a different direction.

The intention of using DULR is so that through DULR the feature map potentially contains perspective change due to the different computing orders, the effect of aggregation for each row is different, thus making it like a change in perspective [23].

2.2. Connectivity preserving centerline dice coefficient (CP_clDice).

The centerline dice coefficient was introduced by Shit *et al.* [24] with the intention of finding what is a good pixels-wise measure to benchmark tubular segmentation while guaranteeing the preservation of the network topology since they are using biomedical images that is brain vascular dataset. They thus proposed the centerline dice coefficient (clDice), as shown in (2) and (3).

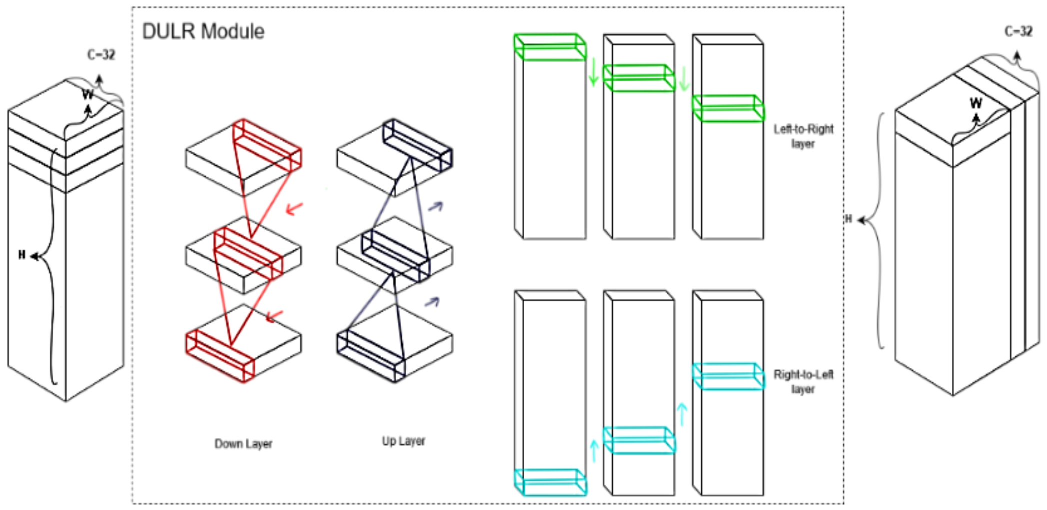


Figure 2. DULR module architecture

$$Tprec(Sp, Vl) = \frac{|Sp \cap Vl|}{Sp} \tag{2}$$

$$Tsens(Sl, Vp) = \frac{|Sl \cap Vp|}{Sl} \tag{3}$$

clDice will create soft skeletonization for the real ground truth mask (Vl) and the predicted mask from the network (Vp). The skeletonization will be called Sl for the skeletonization of the ground truth mask and Sp for the skeletonization of the predicted mask.

The idea of clDice is the computation of soft-Dice, that is to compute the fraction of Sp that lies within Vl, which we call the topology precision ($Tprec$) and the fraction of Sl that lies within Vp, which we call the topology sensitivity ($Tsens$). When $Tprec$ and $Tsens$ have been calculated, it could be considered that $Tprec$ is susceptible to false positives and Tsens to false negative. Since we need to maximize both precision and sensitivity, we define clDice to be harmonic mean also known as F-1 or Dice, the formula is shown in (4) [20].

$$clDice(Vp, Vl) = 2\,x\,\frac{Tprec\ x\ Tsens}{Tprec + Tsens} \tag{4}$$

where $Tprec$ is the topology precision and $Tsens$ the topology sensitivity. Extracting an accurate skeleton is essential to the CP_clDice method. Figure 3 illustrates how Soft-clDice works.

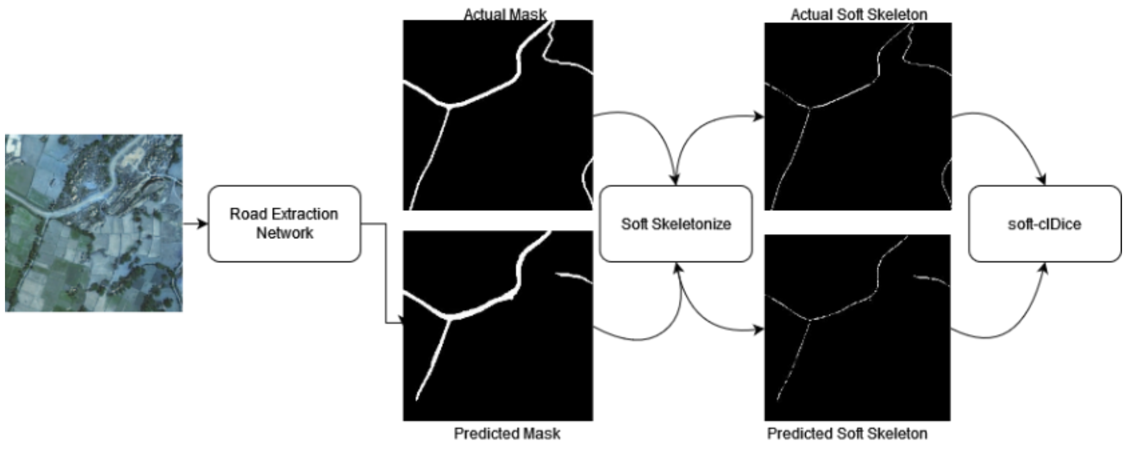


Figure 3. Computation of Soft-clDice

However, skeletonization using erosion and dilation is not fully differentiable and therefore cannot be captured by a loss function. Instead, doing morphological thinning, we use an alternative that is using max-pooling and min-pooling. Thus, a soft-skeletonization technique was proposed to be the alternative, where an iterative min- and max-pooling is applied to approximately simulate erosion and dilation. Figure 4 illustrates how min-pooling, max-pooling, and the use of ReLU activation function could be an alternative.

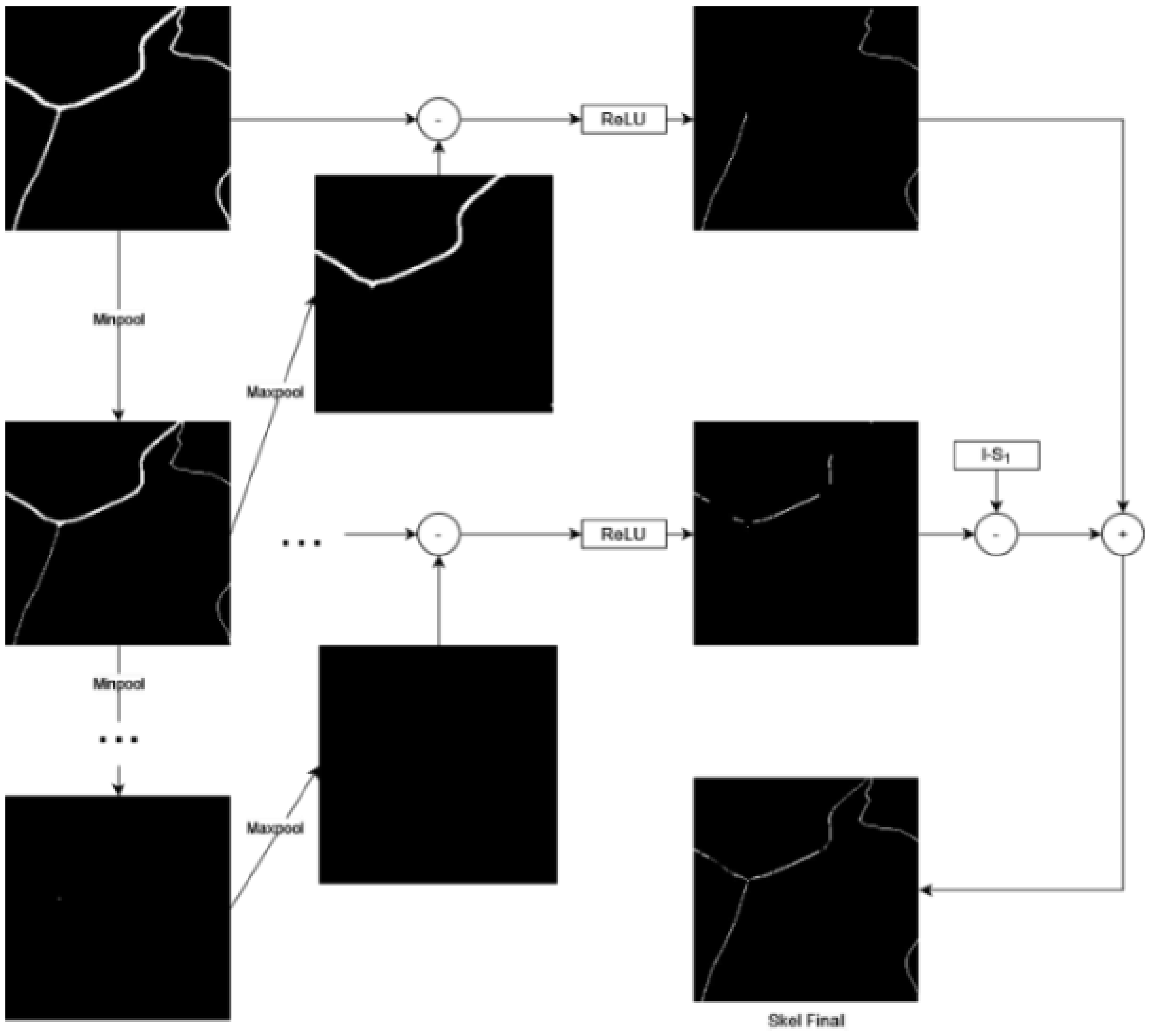


Figure 4. Morphological thinning using min and max pooling

2.3. Proposed method *(*CP_SDUNet*)*

Our proposed method combined SDUNet with the CP_clDice loss function. Previous studies [20] proved that clDice as a loss function improved the segmentation result. Therefore, we use the spatially enhanced extractor in SDUNet to obtain the best road extraction network and preserve its network's topology. Since the objective is to maintain topology while achieving accurate segmentation and not learn the skeletonization, it is proposed that clDice combine with soft-Dice to be used as a loss function represented in (5).

$$L_C = (1 - \alpha)(1 - softDice) + \alpha(1 - softclDice) \quad (5)$$

where α is the coefficient that divides the work for each loss function and a ∈ [0 , 0,5], L_C is the CP_clDice Loss, making this loss function a connectivity preserving the Centerline dice loss function. Figure 5 illustrates what the proposed network architecture looks like.

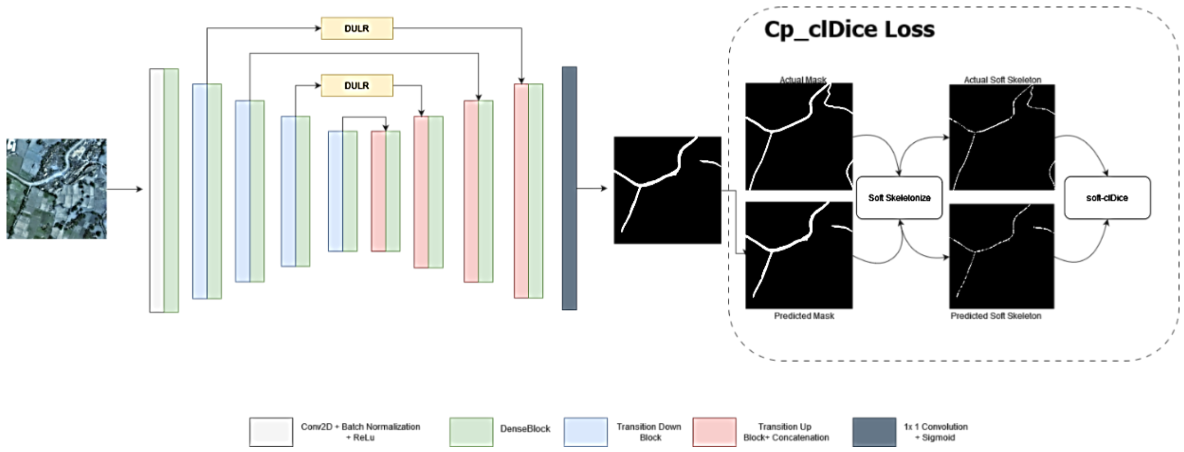


Figure 5. CP_SDUNet architecture

2.4. Experiments

The experiment will be divided into two parts. The first is the CP_SDUNet performance experiment and the joint loss configuration experiment. All the experiments used NVIDIA DGX A-100 GPU with TensorFlow as its framework.

2.4.1. Experiment design

The experimental design used for this research is illustrated in Figure 6. First, for the data preprocessing stage, the input from the dataset will be taken by its original size, then it is cropped and resized to 256×256 pixels or 128×128 pixels. Each size is used in different experiments. After the image is preprocessed, it goes to an augmentation where the data will be increased via augmentation, then to the model for training. After that, the result of each model will be evaluated using the clDice metric and IoU score.

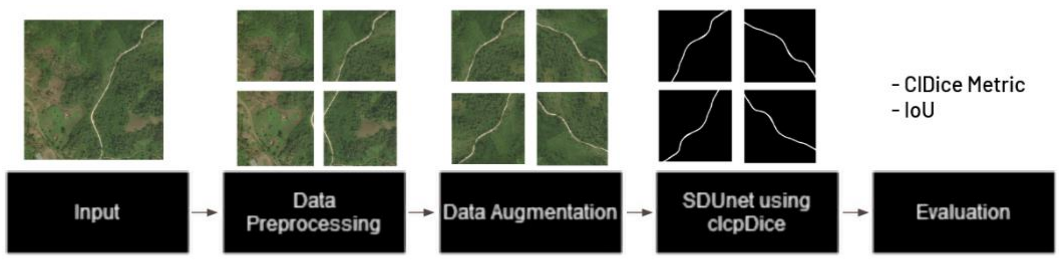


Figure 6. Methods used in this research

2.4.2. Dataset and preprocessing

In these experiments, we used the DeepGlobe road extraction dataset [25]. The DeepGlobe Road Extraction dataset consists of 6,226 aerial images for training, 1,243 validation aerial images, and 1,101 aerial test images. The resolution of each image is 1024×1024 pixels. The DeepGlobe Road Extraction dataset covers images captured over Thailand, Indonesia, and India. The image of the DeepGlobe Road Extraction dataset can be seen in Figure 7, with the example of the real dataset in Figure 7(a) and the image mask in Figure 7(b).

Before the dataset can be used, the dataset will be preprocessed. The images will be first cropped and reduced into four parts. Each part is the same size, 512×512 pixels. After that, the ground truth/mask images will be normalized to a binary image. An example of the preprocessed images is illustrated in Figure 8. The images that have been cropped and resized then need to be processed further with data balancing. The data balancing is intended to remove any insignificant image. The example is illustrated in Figure 9 with the mask image that has significant information explained in Figure 9(a) and the non-significant image in Figure 9(b).

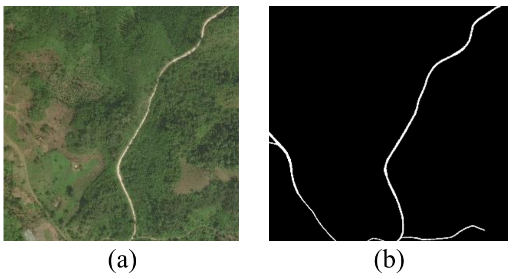


Figure 7. Example image of DeepGlobe dataset (a) real image and (b) mask image

Figure 8. Preprocessed images

The first image is significantly useful because it contains the road that the model needs to learn, and the later image does not have any; thus, it needs to be removed. Not everything but only parts of it; thus, we do data balancing by removing images that only have 0.01% road pixels and below. The distribution of the data before and after balancing can be illustrated in Figure 10, which shows the image distribution before data balancing in Figure 10(a) and after data balancing in Figure 10(b).

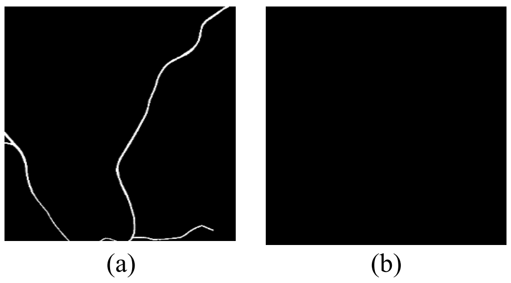


Figure 9. Example kinds of image with (a) information (significant) and (b) no information (insignificant)

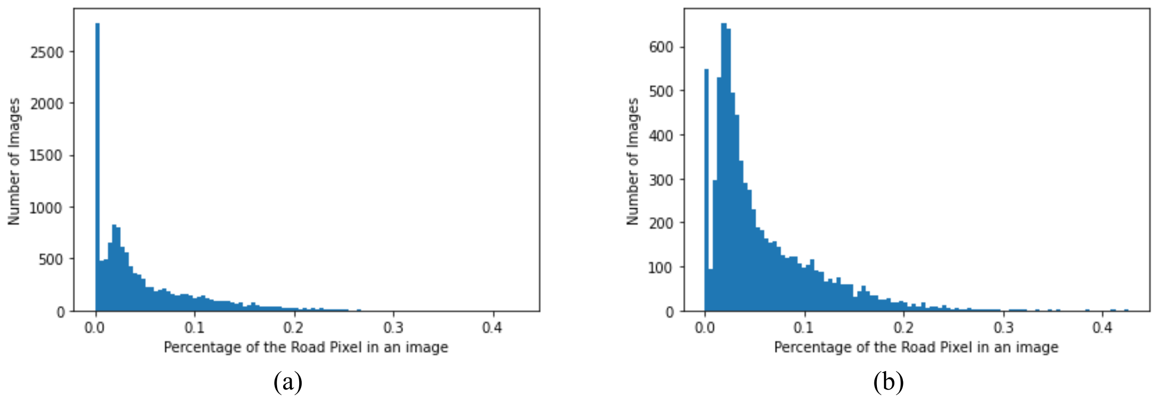


Figure 10. Image distribution (a) before and (b) after data balancing

The next step involves data augmentation to increase the data volume. The augmentation that will be used is random rotation and flipping. After that, the dataset could be used for training but will be used differently for each experiment. The dataset will also vary between half and the whole of the real dataset. Additionally, the dataset will be divided into training, validation, and testing with an 80:10:10 proportion.

2.4.2. Experimental settings and evaluation

We perform the proposed model using the TensorFlow framework. The hyperparameters for the experiment are as follows: The learning rate for training is 0.001, and we also implemented a learning rate scheduler using ReduceLROnPlateau with patience of 4 and a reduced factor of 0.1. The model was trained for 40 epochs, using Adam as its optimizer without weight decay. The proportion for the CP_clDice loss that was used was 90% soft-dice and 10% clDice loss. For all experiments, metrics that are being used to monitor in the training phase are as in (6) to (8).

$$Dice\ Coefficient = 2 \times \frac{TP}{(TP + FP) + (TN + FN)} \tag{6}$$

$$clDice\ Coefficient = 2 \times \frac{Tprec(Sp, Vl) \times Tsens(S, Vp)}{Tprec(Sp, Vl) + Tsens(SL, Vp)} \tag{7}$$

$$IoU = 2 \times \frac{TP}{TP + FP + FN} \tag{8}$$

TP, FP, TN, and FN are sequentially the true positives, false positives, true negatives, and false negatives based on the model prediction and the ground truth/mask.

3. RESULTS AND DISCUSSION

3.1. CP_SDUNet performance

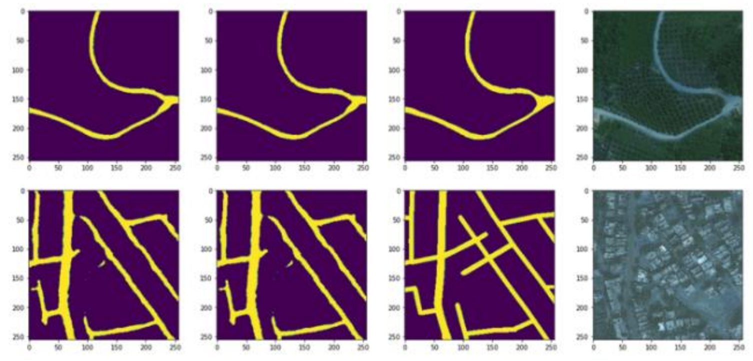
The main experiment is intended to evaluate SDUNet trained by various loss functions for road extraction, one of which is the proposed method CP_SDUNet. This experiment is implemented using 256×256 pixels images and 50% of the actual data for training the model. Due to the size of the data, we used half of the actual data in this experiment. The result from the experiment is shown in Table 1.

Table 1. The training, validation, and testing result of SDUNet for its loss function variations

Value	Training		Validation		Testing	
	SDUNet using BCE Loss	SDUNet using CP_clDice Loss (CP_SDUNet)	SDUNet using BCE Loss	SDUNet using CP_clDice Loss (CP_SDUNet)	SDUNet using BCE Loss	SDUNet using CP_clDice Loss (CP_SDUNet)
IoU	0.5303	**0.61**	0.4422	**0.5781**	0.52	**0.593**
clDice	0.7750	**0.8293**	0.6964	**0.8096**	0.308	**0.88**
Loss	**0.0581**	0.2283	**0.1225**	0.2522	-	

The result given in Table 1 shows that SDUNet using BCE loss performed worse compared to SDUNet using CP_clDice loss for its training, validation, and testing results. With the CP_clDice loss, the model could achieve better performance with an interest over union (IOU) score of 0.61 compared to BCE Loss with an IoU score of 0.53. A 0.08 difference resulted in a very distinct outcome.

The validation phase also had a big difference, with the CP_clDice loss achieving an IoU score of 0.5781, and the BCE loss obtaining an IoU score of 0.44. A whole 0.13 difference between the two. Hence, the testing result that was shown in Table 1 further confirmed that SDUNet using CP_clDice (CP_SDUNet) loss was able to outperform SDUNet using BCE Loss with an IoU score of 0.59 for CP_clDice loss and an IoU score of 0.52 for BCE loss, a distinct 0.7 difference. Interestingly, even though BCE loss generally produces a good IoU Score, SDUNet with BCE loss obtained a very low clDice score, which means that the model could create extraction correctly but did not give a very good centerline for its road. The result of the two models is shown in Figure 11, which shows the result of the predicted road when SDUNet is trained using clDice loss in Figure 11(a) and the result of the predicted road when SDUnet is trained using BCELoss in Figure 11(b).


(a)

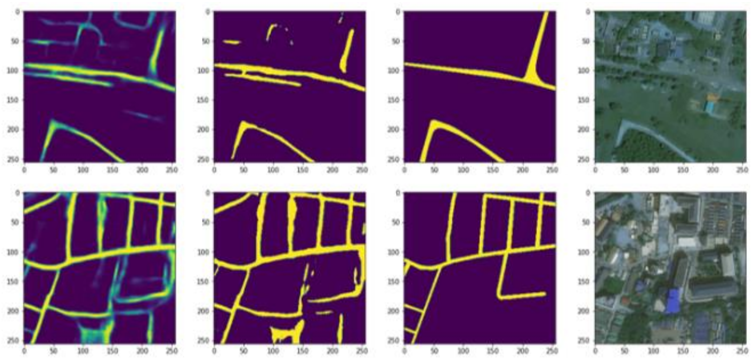

(b)

Figure 11. Result of the predicted road from testing data (a) SDUNet using clDice and (b) SDUNet using BCE Loss

Next, training with only half of the dataset and a whole dataset would result in a different outcome. However, the problem with using the whole dataset is that the data needs to be transformed into an Image with a size of 128×128 pixels so that it does not exceed the memory limit, which may lead to the loss of its details. Hence, we conducted an experiment using 128×128 pixels input with the whole dataset as its input.

The experiment was intended to test CP_SDUNet using the whole of the DeepGlobe Road Extraction dataset by reducing the input image size to 128×128 pixels. Thus, the result is shown in Table 2 and Figure 12.

Table 2. The training, validation and testing result of CP_SDUNet using image size of 128×128 pixels

Value	Training	Validation	Testing
clDice	0.8049	0.7767	0.729
IoU	0.6041	0.5502	0.578
Loss	0.2433	0.2841	-

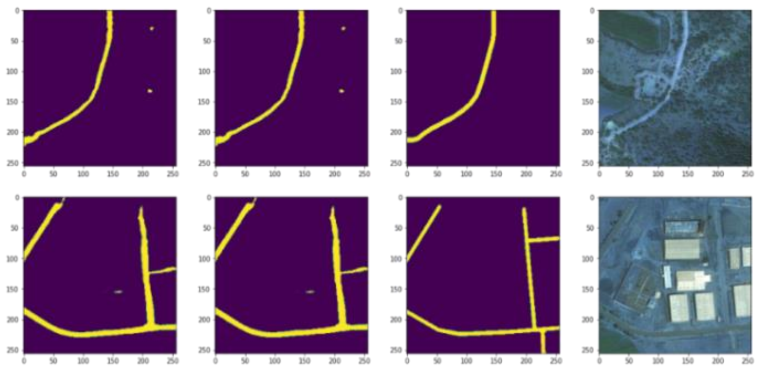


Figure 12. Testing data result using SDUNet with image size of 128×128 pixels

From what we achieved, the result showed that using the whole dataset does give a better result, but not that significant. The testing result score for clDice decreases from 0.738 to 0.729 while the IoU score increases to 0.578. From the testing results, we could also see that the model has successfully extracted the roads, but it is still confusing since the image became much smaller. Many pixels that were supposed to support the learning were either blurred or removed because of the resizing. Thus, using the whole dataset while lowering the resolution leads to suboptimal changes.

3.2. Joint loss variation performance

We have tested CP_SDUNet using the combination of 90% Dice loss and 10% clDice loss as its loss function configuration. Here, we want to experiment with other combinations to find the best combinations for road extraction using CP_clDice.

3.2.1. 80% Dice Loss and 20% clDice loss

The input image is set to be 128×128 pixels. The batch size and other hyperparameters are the same configuration as the previous experiments. The result is shown in Table 3 and Figure 13.

Table 3. Training, validation and testing result for SDUNet using 80% dice loss and 20% clDice loss

Value	Training	Validation	Testing
clDice	0.8068	0.7943	0.85
IoU	0.5923	0.5585	0.65
Loss	0.2439	0.2683	-

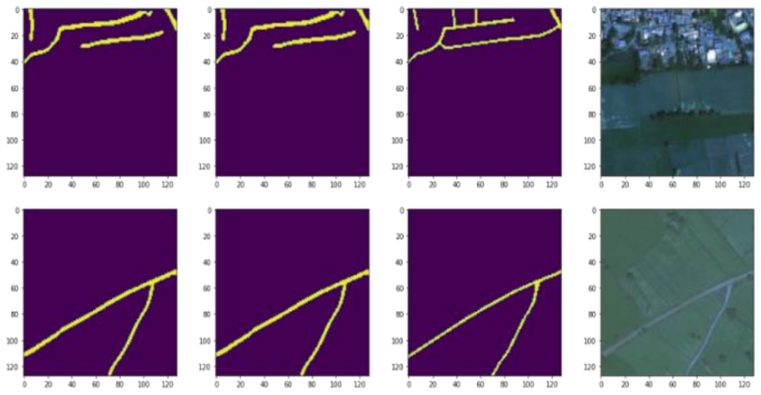


Figure 13. Testing data result for CP_SDUNet using the combination of 80% dice loss and 20% clDice loss

The results were better than using the 10% combination where the testing score of clDice is 0.85, and the IoU is 0.65, which is the best IoU so far, meaning the model could predict roads well. From the testing result images, we could also see that it is almost perfect on the second image, while for the first image, the model still could not perfectly extract roads. This is due to the side effect of resizing input images.

3.2.2. 70% Dice Loss and 30% clDice loss

Knowing that the combination of 20% clDice loss and 80% dice loss gave a better result, we tried to even further increase the proportion for clDice loss, with 30% and 70% being dice loss. The result of this experiment is described in Table 4 and Figure 14.

The result indicates that increasing clDice loss proportion to 30% led to a poor performance. The model started to learn only one part: that is the road. The result may be due to the difficulty of learning the road's location because of clDice loss. clDice loss uses morphology to make skeletonization, and because the factor has become greater, it is harder for the model to discern which part is the road. Thus, the experiment showed that increasing alpha for clDice is not always good for the model, and the best fit for SDUNet is the combination of 20% clDice loss and 0.8 dice loss.

Table 4. Training, validation and testing result for SDUNet using 70% dice loss and 30% clDice loss

Value	Training	Validation	Testing
clDice	0.0681	0.0690	0.08
IoU	0.0355	0.0357	0.03
Loss	0.9317	0.9312	-

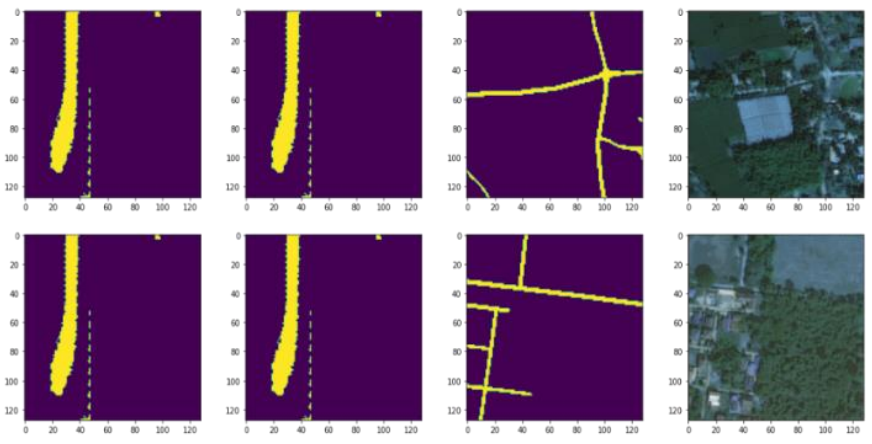


Figure 14. Testing data result for CP_SDUNet using the combination of 70% dice loss and 30% clDice loss

4. CONCLUSION

The experiment result indicates that the proposed method, CP_SDUNet, achieved excellent road extraction due to its spatial context enhancement and further improved with the help of the CP_clDice loss. The CP_clDice loss improves the performance and convergence speed for road extraction tasks. Besides that, CP_clDice outperforms BCE loss significantly, providing greater confidence on the segmenting road and converging much faster. CP_clDice is influenced by its joint loss function, in which CP_clDice performs the best when the dice score is prioritized over the centerline dice score and used as complementary. From the experiment, we can conclude that the best-performing model configuration uses CP_SDUNet with a joint loss configuration of 20% clDice and 80 % dice loss while having 128×128 pixels as the image input size. The model yields an IoU score of 0.65 and a clDice score of 0.85, which is the best result throughout the experiment. Our findings underscore the importance of preserving road connectivity for practical applications. The results provide strong empirical evidence supporting CP_SDUNet's potential as a reliable solution for road extraction from satellite imagery.

ACKNOWLEDGEMENTS

The authors express gratitude to the UI AI Center of Excellence, Faculty of Computer Science, Universitas Indonesia, for providing access to the NVIDIA DGX-A100 system necessary to run the experiments.

FUNDING INFORMATION

This study was conducted without formal funding or research grants.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Bayu Satria Persada	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	
Muhammad Rifqi Priyo Susanto	✓		✓				✓	✓	✓					
Laksmita Rahadianti	✓	✓		✓	✓	✓				✓	✓	✓	✓	
Aniati Murni Arymurthy	✓			✓						✓		✓		

C	: **C**onceptualization	I	: **I**nvestigation	Vi	: **Vi**sualization
M	: **M**ethodology	R	: **R**esources	Su	: **Su**pervision
So	: **So**ftware	D	: **D**ata Curation	P	: **P**roject administration
Va	: **Va**lidation	O	: Writing - **O**riginal Draft	Fu	: **Fu**nding acquisition
Fo	: **Fo**rmal analysis	E	: Writing - Review & **E**diting		

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY

- The details of the data and its availability were originally publicly available in IEEE Explore at http://doi.org 10.1109/CVPRW.2018.00031.
- The data used in this study is publicly available and can be accessed from Deep Globe Road Extraction Dataset on Kaggle. The Dataset can be found at the following link: DeepGlobe Road Extraction Dataset (https://www.kaggle.com/datasets/balraj98/deepglobe-road-extraction-dataset).

REFERENCES

[1] M. Naouai, A. Hamouda, and C. Weber, "Urban road extraction from high-resolution optical satellite images," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6112 LNCS, no. PART 2, pp. 420–433, 2010, doi: 10.1007/978-3-642-13775-4_42.

[2] Z. Chen *et al.*, "Road extraction in remote sensing data: A survey," *International Journal of Applied Earth Observation and Geoinformation*, vol. 112, no. March, 2022, doi: 10.1016/j.jag.2022.102833.

[3] N. Sun, J. X. Zhang, G. M. Huang, Z. Zhao, and L. J. Lu, "Review of road extraction methods from SAR image," *IOP Conference Series: Earth and Environmental Science*, vol. 17, no. 1, 2014, doi: 10.1088/1755-1315/17/1/012245.

[4] N. Sun *et al.*, "Road network extraction from SAR images with the support of angular texture signature and POIs," *Remote Sensing*, vol. 14, no. 19, 2022, doi: 10.3390/rs14194832.

[5] F. Bastani *et al.*, "RoadTracer: Automatic extraction of road networks from aerial images," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4720–4728, 2018, doi: 10.1109/CVPR.2018.00496.

[6] M. Lan, Y. Zhang, L. Zhang, and B. Du, "Global context based automatic road segmentation via dilated convolutional neural network," *Information Sciences*, vol. 535, pp. 156–171, 2020, doi: 10.1016/j.ins.2020.05.062.

[7] M. J. Khan and P. P. Singh, "Advanced road extraction using CNN-based U-net model and satellite imagery," *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, vol. 5, p. 100244, Sep. 2023, doi: 10.1016/j.prime.2023.100244.

[8] E. Tsalera, A. Papadakis, M. Samarakou, and I. Voyiatzis, "Feature extraction with handcrafted methods and convolutional neural networks for facial emotion recognition," *Applied Sciences*, vol. 12, no. 17, Aug. 2022, doi: 10.3390/app12178455.

[9] R. Lian and L. Huang, "DeepWindow: Sliding window based on deep learning for road extraction from remote sensing images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, no. d, pp. 1905–1916, 2020, doi: 10.1109/JSTARS.2020.2983788.

[10] F. Zohourian, B. Antic, J. Siegemund, M. Meuter, and J. Pauli, "Superpixel-based road segmentation for real-time systems using CNN," *VISIGRAPP 2018 - Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, vol. 5, no. Visigrapp, pp. 257–265, 2018, doi: 10.5220/0006612002570265.

[11] Y. Liu *et al.*, "A road extraction method based on region growing and mathematical morphology from remote sensing images," *Journal of Computer and Communications*, vol. 06, no. 11, pp. 91–97, 2018, doi: 10.4236/jcc.2018.611008.

[12] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018, doi: 10.1109/TPAMI.2017.2699184.

[13] Y. Li *et al.*, "Learning dynamic routing for semantic segmentation," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 8550–8559, 2020, doi: 10.1109/CVPR42600.2020.00858.

[14] Y. Wei, K. Zhang, and S. Ji, "Road network extraction from satellite images using CNN based segmentation and tracing," *International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 3923–3926, 2019, doi: 10.1109/IGARSS.2019.8898565.

[15] Y. Li, R. Zhang, and Y. Wu, "Road network extraction in high-resolution SAR images based CNN features," *International Geoscience and Remote Sensing Symposium (IGARSS)*, vol. 2017-July, pp. 1664–1667, 2017, doi: 10.1109/IGARSS.2017.8127293.

[16] W. Weng and X. Zhu, "UNet: Convolutional networks for biomedical image segmentation," *IEEE Access*, vol. 9, pp. 16591–16603, 2021, doi: 10.1109/ACCESS.2021.3053408.

[17] M. Yang, Y. Yuan, and G. Liu, "SDUNet: Road extraction via spatial enhanced and densely connected UNet," *Pattern Recognition*, vol. 126, pp. 1–8, 2022, doi: 10.1016/j.patcog.2022.108549.

[18] H. He, D. Yang, S. Wang, S. Wang, and Y. Li, "Road extraction by using atrous spatial pyramid pooling integrated encoder-decoder network and structural similarity loss," *Remote Sensing*, vol. 11, no. 9, pp. 1–16, 2019, doi: 10.3390/rs11091015.

[19] A. Abdollahi, B. Pradhan, and A. Alamri, "VNet: An end-to-end fully convolutional neural network for road extraction from high-resolution remote sensing data," *IEEE Access*, vol. 8, pp. 179424–179436, 2020, doi: 10.1109/ACCESS.2020.3026658.

[20] A. Abdollahi, B. Pradhan, and A. Alamri, "SC-RoadDeepNet: A new shape and connectivity-preserving road extraction deep learning-based network from remote sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–15, 2022, doi: 10.1109/TGRS.2022.3143855.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," Jul. 2017. doi: 10.1109/cvpr.2017.243.

[23] J. Gao, Q. Wang, and X. Li, "PCC Net: Perspective crowd counting via spatial convolutional network," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 10, pp. 3486–3498, 2020, doi: 10.1109/TCSVT.2019.2919139.

[24] S. Shit *et al.*, "CLDICE - A novel topology-preserving loss function for tubular structure segmentation," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 16555–16564, 2021, doi: 10.1109/CVPR46437.2021.01629.

[25] I. Demir *et al.*, "DeepGlobe 2018: A challenge to parse the earth through satellite images," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2018-June, pp. 172–181, 2018, doi: 10.1109/CVPRW.2018.00031.

BIOGRAPHIES OF AUTHORS

Bayu Satria Persada received the B.Eng degree in computer engineering from Universitas Indonesia in 2021. He is currently a master's student at the Faculty of Computer Science, Universitas Indonesia, Depok, West Java. He is someone who is full of confidence, diligent, honest, keen and able to work under pressure. Currently, he indulges himself in learning computer vision as his main research, but also has interests in machine learning, deep learning, and artificial intelligence. He can be contacted at bayu.satria31@ui.ac.id.

Muhammad Rifqi Priyo Susanto the B.Cs. degree from Universitas Negeri Surakarta Sebelas Maret, in 2022. He is currently pursuing his master's degree at the Faculty of Computer Science, University of Indonesia. His areas of interest are in digital image processing, machine learning, image segmentation, and remote sensing He can be contacted at muhammad.rifqi39@ui.ac.id.

Laksmita Rahadianti received the B.CS. degree from University of Indonesia in 2009, the M.Sc. degree in Color in Informatics and Media Technology from Jean Monnet University, France in 2012, and the Ph.D. degree in Computer Science and Engineering from Nagoya Institute of Technology in 2018. At the moment, she is an assistant professor at the Faculty of Computer Science, Universitas Indonesia. Her research interests include computer vision, image processing, photometry, and perception. She can be contacted at laksmita@cs.ui.ac.id.

Aniati Murni Arymurthy received the Ir. degree from Universitas Indonesia, M.Sc. degree from Department of Computer and Information Sciences, The Ohio State University and her Ph.D. degree from Universitas Indonesia. Currently she is a professor at the Faculty of Computer Science, Universitas Indonesia with specialization in image processing and spatial data analysis. She can be contacted at aniati@cs.ui.ac.id.