Received: August 4, 2022. Revised: August 31, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

BVU-Net: A U-Net Modification by VGG-Batch Normalization for Retinal Blood Vessel Segmentation

Anita Desiani[1] Erwin[2*] Bambang Suprihatin[1] Yogi Wahyudi[1]
Endro Setyo Cahyono[1] Muhammad Arhami[3]

[1] *Mathematics Department, Mathematics and Natural Science Faculty, Universitas Sriwijaya, Indralaya, Indonesia*
[2] *Computer Engineering Department, Computer Science Faculty, Universitas Sriwijaya, Indralaya, Indonesia*
[3] *Informatics Technique Department, Politeknik Negeri Lhokseumawe, Lhokseumawa, Indonesia*
* Corresponding author's Email: erwin@unsri.ac.id

Abstract: The study proposes a BVU-Net architecture that combines the advantages of VGG and U-Net. The VGG architecture has a smaller kernel size which speeds up the training process. VGG bears some resemblance to the Encoder portion of U-Net. In this study, the BVU-Net encoder uses the VGG architecture with the addition of batch normalization. The addition of batch normalization aims to help simplify weight initialization so that the training process is faster and reduces the risk of overfitting, while the decoder section still uses the U-Net architecture. BVU-Net is expected to be able to overcome the weakness of U-Net architecture in retinal image blood vessel segmentation. The performance results produced by BVU-Net on the DRIVE dataset are 96.36% accuracy, 78.71 sensitivity, 98.1% specificity, F1-score 78.9, and IOU 0.65. The results of BVU-Net performance on the STARE dataset are accuracy of 96.39%, sensitivity of 79.35%, specificity of 98.52%, F1-score of 77.16%, and IoU of 0.63. Based on these results indicate that BVU-Net has a better performance on the DRIVE dataset in detecting retinal blood vessels than STARE. This can be seen from the sensitivity value of DRIVE which is higher than STARE. The BVU-Net IoU results on DRIVE and STARE are greater than 0.5 and the F1-score above 70%, indicating that BVU-Net is capable and balanced in detecting the intersection area between blood vessels and the background on retinal images.

Keywords: CNN method, Retinal blood vessels, Segmentation, U-Net architecture, VGG architecture.

1. Introduction

The retina is consisted of millions of nerve cells or photoreceptors that react to light [1]. The structure of the retinal blood vessels can help detect Diabetic Retinopathy (DR) [2]. So far, the detection of retinal blood vessels has been done manually by ophthalmologists through retinal images taken from the fundus camera. The image results obtained usually have reasonably low image quality, and there is still noise, making it difficult for ophthalmologists to detect diseases of the retina [3]. In this case, an automatic retinal diagnosis system with the help of a computer is needed, which can be developed to assist ophthalmologists in conducting a more efficient and accurate retinal diagnosis, namely retinal blood vessel segmentation [4].

Convolutional neural network (CNN) is one of the segmentation methods with strong capabilities when trained with large datasets [5]. The primary ability of CNN lies in its architecture, where the architecture that is often used in biomedical segmentation is U-Net [6]. U-Net is a U-shaped architecture consisting of two paths, namely encoder (contract layer) and decoder (extension layer). The encoder path process is used to reduce the size of the input matrix by increasing the number of feature maps. In contrast, the decoder path returns the matrix to its original size by minimizing the number of feature maps so that the image can be segmented [7]. The study in [8] used U-Net in retinal blood vessel segmentation with the DRIVE dataset. It yielded 94% accuracy, 72% sensitivity, and 97% specificity. Other research by Fu et al in [9] with the DRIVE dataset delivered a

sensitivity value of 72% and an accuracy of 94%, while with the STARE dataset, it was 71% and 95%. Other research [10] with the DRIVE dataset yielded 95% accuracy, 73% sensitivity, and 94% specificity. The three studies did not calculate other performance such as the Intersection over Union (IoU) and F-1 score. In addition, the sensitivities were also relatively low. The U-Net architecture has a shallow learning layer which sometimes causes learning on features to be suboptimal. This causes the sensitivity generated by U-Net in some studies to be low. The addition of layers to the U-Net architecture can add to the complexity of the U-Net network so that the execution time required is long and the inability to run on some computers with limited specifications [7].

One way to address the problem of U-Net is by combining different architectures. The visual geometry group (VGG) encoder path has a similar structure to U-Net but with a deeper layer. Despite having a deeper layer, VGG uses a smaller kernel to help speed up training times [7]. Several studies that modified the U-Net and VGG architectures: the study in [11] proposed the VGG-16 and U-Net architectures in the brain segmentation process resulting in F1-score values above 90%. Another study by [12] proposed the VGG and U-Net architecture for ultrasonic image detection but it produced PSNR and SSIM values of 39 dB and 0.99, respectively. Another study in [13] proposed a U-Net architecture with a VGG encoder in the multi-class segmentation of heart images resulting in an F1-score of 94.3%.

The VGG architecture layer belongs to a very deep layer so it can produce a large number of parameters [14]. The deeper the layer, the more parameters are used and the more difficult it is to initialize the learning weights. Batch normalization is a regularization technique used to assist the learning process. although it does not significantly help overcome overfitting such as dropout, batch normalization can reduce internal covariance changes and avoid instability of activation distribution in deep network layers[15]. on deeper networks with many parameters, weights are difficult to initialize. Batch normalization makes weights easier to initialize. It allows for a much higher learning rate thereby increasing the speed of the training network and reducing the risk of overfitting[15]. Batch normalization performs a normal distribution by adjusting the distribution average and the input layer variance [14]. Research in [16] proposed a U-Net architecture by adding a batch normalization layer to brain tumor segmentation resulting in an average F1-score of 86%. The study in [17] proposed the DCNN U-Net architecture by adding batch normalization to the segmentation of skin lesions resulting in 93% accuracy, 82.9% sensitivity, 98.9% specificity, 75.2% IoU, and 84% F1-score. Several studies have shown that adding batch normalization to CNN architectures such as U-Net and other architectures can improve model performance and speed up the training process.

This study introduces a new architecture, namely BVU-Net to overcome weaknesses in U-Net and improve U-Net performance results. BVU-Net architecture is a combination of U-Net and VGG with the addition of batch normalization. VGG on BVU-Net is implemented in the encoder part of U-Net to handle the activation distribution instability due to multiple layers and parameters. Batch normalization is added to each convolution layer on the encoder with the aim of making it easier to initiate learning weights and speed up the training process even though the learning layer is deeper. The proposed architecture is expected to be a robust architecture in blood vessel segmentation on retinal image. DRIVE and STARE datasets are used in this study to see the superior performance of BVU-Net based on the results of the accuracy, sensitivity, specificity, F1-score obtained.

The structure of the paper is organized as follows: section 2 presents the proposed method, which includes collecting data, image enhancement, the patching technique, BVU-Net architecture, training model, testing data, and evaluation. Section 3 describes the results and discussion. Section 4 concludes the paper.

2. Methods

The study has several steps, namely pre-processing, architectural modification, training, testing, and evaluation (Fig. 1).

2.1 Data collection

The data used in this study is the digital retinal images for vessel extraction (DRIVE) and structured analysis of the retina (STARE) datasets. The DRIVE is freely available at https:// www.isi.uu.nl/Research/Databases/DRIVE/ with 40 retinal image samples taken at random from 400 people with diabetes aged between 25 and 90 years in the Netherlands. This dataset is divided into 20 training images and 20 test images. In comparison, the STARE dataset obtained from the page https://cecas.clemson.edu/~ahoover/stare was a research project in 1975 at the university of California. This dataset consists of 400 retinal images,

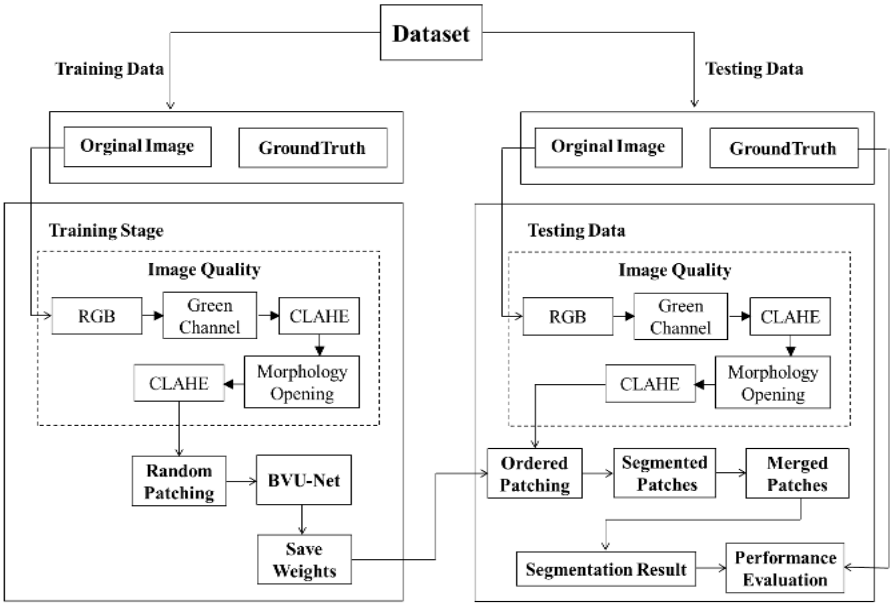


Figure. 1 Workflow diagram of the proposed retinal blood vessel segmentation

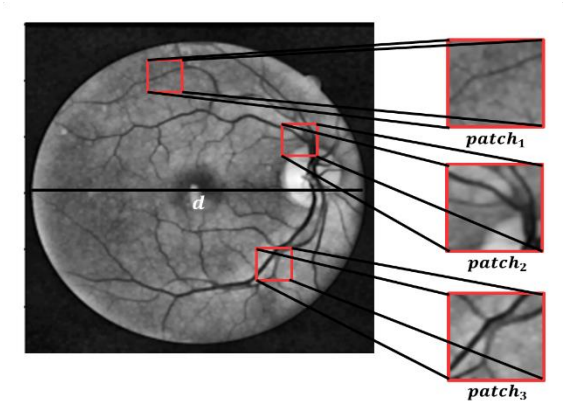


Figure. 2 Illustration of patching technique in the training process for retinal Image on STARE dataset

of which 20 images have retinal blood vessel ground truth.

2.2 Image enhancement

Retinal imagery is an image enhancement process by first taking one of the channels in RGB. The green channel is one of the RGB channels chosen because it has high intensity and contrast in retinal blood vessels [1]. The mathematical equation used to obtain the green channel is Eq. (1) [18].

$$g = \frac{G}{(R+G+B)} \tag{1}$$

where g is the green channel, R is the red component, G is the green component, and B is the blue component.

The results of the green channel are improved image quality using contrast limited adaptive histogram equalization (CLAHE). CLAHE is applied to images that have low contrast by dividing them into two parameters, namely block size (BS) and clip limit (CL). These two parameters control the image quality to be improved [18]. The histogram equation for each block can be defined as in Eq. (2) [18].

$$N_{avg} = \frac{N_{rX} \times N_{rY}}{N_{gray}} \tag{2}$$

where N_{avg} is the average number of pixels, N_{gray} is the number of grey levels, N_{rX} and N_{rY} is the number of pixels in the X and Y dimensions, respectively. Perform CL calculations using Eq. (3) [18].

$$N_{CL} = N_{clip} \times N_{avg} \tag{3}$$

where N_{CL} is the actual CL value, N_{clip} is the CL input value in the range $[0, 1]$. If the number of pixels is greater than N_{CL}, the pixels will be truncated.

The results of CLAHE are contour smoothing and the removal of thin objects in the image using morphology opening. The morphological opening equation is obtained from the erosion operation first and then the dilation operation. The morphological opening equation can be defined as Eqs. (4), (5), and (6) [18].

$$A \circ B = (A \ominus B) \oplus B \tag{4}$$

$$A \oplus B = \{x|(B)_x \cap A \neq \phi\} \tag{5}$$

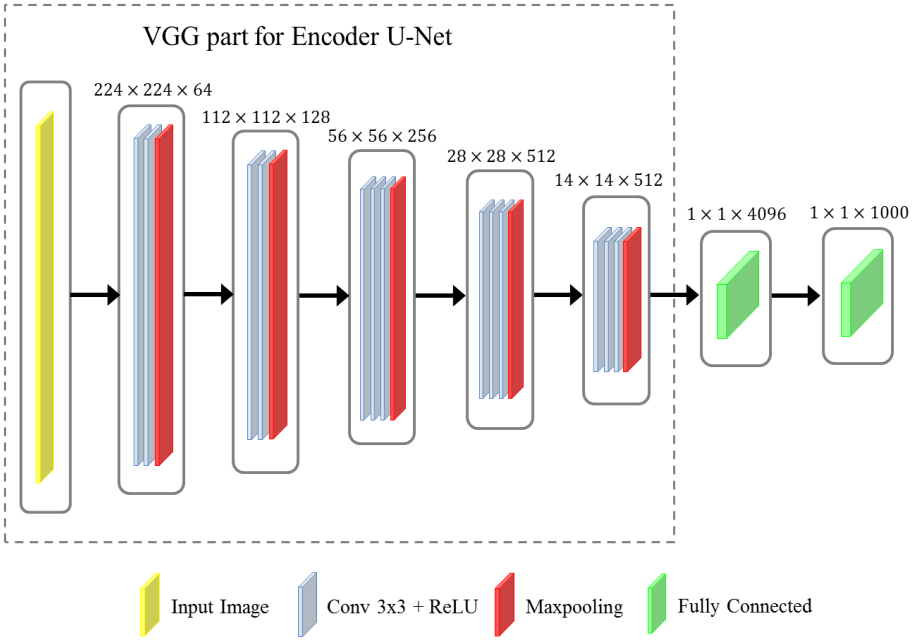


Figure. 3 VGG architecture for classification problem

$$A \ominus B = \{x|(B)_x \cap A^C \neq \phi\} \quad (6)$$

where A is the original image, B is an element structure or operator matrix $\ominus$ is an erosion operation, and $\oplus$ a dilation operation.

2.3 Image patching

The DRIVE and STARE datasets consist of 20 and 40 training images, respectively. Generally, the performance of the CNN architecture is strongly influenced by a large amount of training data. Patching is a technique used to reproduce data. In training data patching works by dividing the image into small pieces randomly of the same size. While in the testing process, the patching technique is done by dividing the image into small pieces sequentially starting from the pixels at the starting point of the image with the same size as the patches on the training image. Fig. 2 illustrates the patching method utilized during the training procedure.

As illustrated in Fig. 2, the patching approach used throughout the training phase involves plucking small, circular-shaped chunks from the retinal picture. As shown, there are three samples of patches that were chosen at random along the retinal image's diameter (d) and were the same size.

2.4 BVU-Net architecture

BVU-Net architecture is modified using the VGG and the U-Net architecture. U-Net architecture is used for segmentation dan consists of two parts, namely the encoder and the decoder [6]. The VGG is a CNN architecture commonly used for classification consisting of 5 convolution blocks and 3 fully connected layers. Each block in VGG includes a 3×3 convolution layer and max pooling. In the fifth block, the output matrix (flatten) is evenly distributed which produces two layers for classification.The VGG architecture, can be seen in Fig. 3. The VGG architecture is similar to the encoder on the U-Net architecture, where the difference is only in the number of convolutions in the 3rd to 5th block. The addition of a batch normalization process to each convolution block in the encoder path is to facilitate the initiation of weights in learning. The proposed BVU-Net architecture can be seen in Fig. 4.

2.5 Training model

The initial process carried out on the training data is to initialize the parameters used, such as the number of epochs and batch size. The next step is that the image input is divided into 60% training data and 40% validation data before entering the training data process using architectural modifications. In this architecture, in the encoder path, a convolutional layer process will be carried out using Eq. (7) [12].

$$A_q^p = (C_p * K_q) + b_q \quad (7)$$

where * represents the convolution operation, where if the convolution operation is performed on each matrix entry, it will obtain Eqs. (8) and (9).

$$c_{i,j} * k_{i,j} = \left(\sum_{u=0}^{n-1}\sum_{v=0}^{n-1}(c_{u+i,v+j} \times k_{u+1,v+1})\right) \quad (8)$$

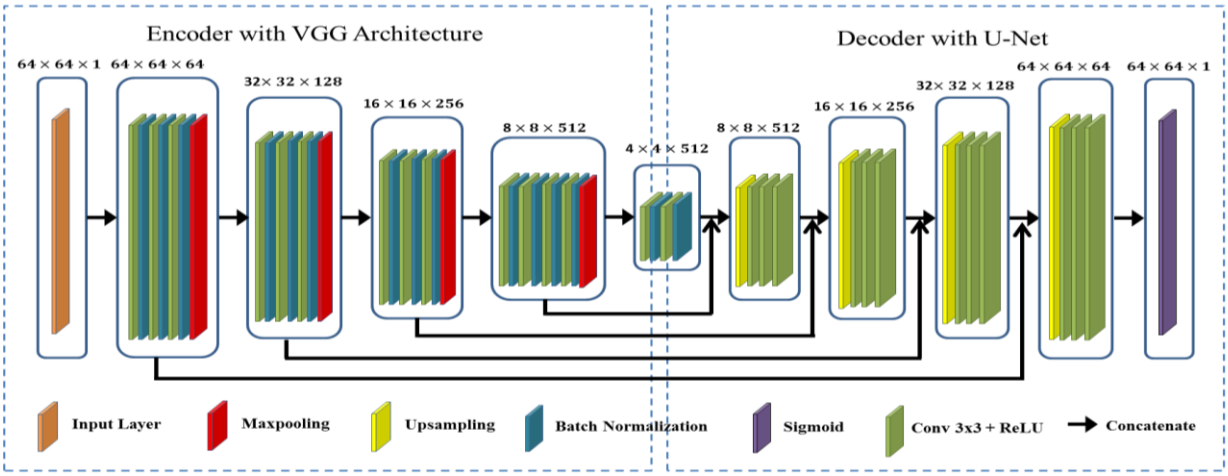


Figure. 4 The BVU-Net architecture; modification of U-Net and VGG architecture with addition of batch normalization on each convolution layer

$$a_{i,j} = \left(\sum_{u=0}^{n-1}\sum_{v=0}^{n-1}\left(c_{u+i,v+j} \times k_{u+1,v+1}\right)\right) + b_q \quad (9)$$

where i is the row, j is the column, n is the kernel height measurement, p is the number of input, q is the number of filters, A_q^p is the matrix of a result of q-th convolution (feature maps) on p-th input, $a_{i,j}$is the i-th row entry of the j-th column in the matrix A_q^p, $C_p$is the input matrix, $c_{i,j}$ is the j-th column of the i-th row entry of the $C_p$matrix, $K_q$is the q-th kernel matrix, $k_{i,j}$is the i-th row entry of the j-th column in the K_q matrix, and b_q is the q-th bias.

The results of the convolutional layer process are calculated using the ReLU activation function using Eq. (10) [19].

$$r(z) = max(z, 0) = \begin{cases} z & if\ z \geq 0 \\ 0 & if\ z < 0 \end{cases} \quad (10)$$

where $r(z)$ is the ReLU output, and z is the activation function input.

Each matrix entry obtained from the calculation of the ReLU activation function will be normalized with batch normalization using Eqs. (11), (12), and (13) [5].

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij} \quad (11)$$

$$\sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}\left(x_{ij} - \mu_j\right)^2 \quad (12)$$

$$\hat{x}_{ij} = \frac{x_{ij} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}} \quad (13)$$

where $\mu_j$is the average for each mini-batch (column), $\sigma_j^2$is the variance for each mini-batch, j is the number of mini-batches, m is the amount of data in one mini-batch, $\hat{x}_{ij}$ is the normalized matrix entry, x_{ij} is the matrix entry input in the i-th row and j-th column, and ε is the smallest positive constant value.

The result of the batch normalization process is that the dimensions of feature maps are reduced by max-pooling 2×2 and the dimensions of feature maps are increased by upsampling of 2×2. The matrix results from the convolution process on the encoder path with the decoder path are combined using concatenate. Do the convolutional process again, only the size used is 1×1 by applying a sigmoid activation layer using Eq. (14) [19].

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (14)$$

where $z \in (-\infty, \infty), \sigma(z) \in (0, 1)$.

The final step is to calculate the loss function: binary cross entropy using Eq. (15) [5].

$$L = -\frac{1}{m \times n}\left[\sum_{i=1}^{m}\sum_{j=1}^{n}\begin{pmatrix}\left(y_{i,j}\ log\left(p_{i,j}\right)\right) + \\ \left(\left(1 - y_{i,j}\right) log\left(1 - p_{i,j}\right)\right)\end{pmatrix}\right] \quad (15)$$

where m is the number of rows of pixels, n is the number of columns of pixels, p is the predicted probability value and$y$ is the ground truth value (0 for background pixels and 1 for blood vessel pixels). The results of the best result of training are stored as weights in the model.

2.6 Testing data

The segmentation prediction results will be contained in the confusion matrix thanks to testing done using 20 test data from DRIVE and STARE to test the model that has been trained on those data. To

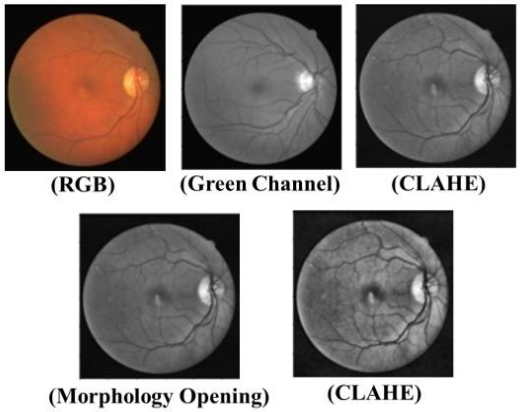


Figure. 5 Example of pre-processing results in the "25_training.tif" file in DRIVE dataset

segment retinal blood vessels, the confusion matrix is separated into two labels: label 1 for pixels that are retinal blood vessels (foreground) and label 0 for pixels that are not retinal vessels (background).

2.7 Evaluation

At this stage, the accuracy, sensitivity, specificity, F1-score, intersection over union (IoU), and receiver operating characteristics (ROC) curve values will be used to calculate the performance evaluation of the model. Results of the model's performance were compared with those from other studies.

3. Result and discussion

3.1 Image enhancement and patching

At this stage of image enhancement quality, it aims to improve data quality for the better. The initial image used for this stage is obtained from the DRIVE and STARE datasets. The results of the image enhancement can be seen in Fig. 5 for DRIVE dataset.

In Fig. 5, it can be seen that the image input in each dataset of type BGR is converted to RGB and followed by taking one of the color channels, namely the green channel. The results of the green channel are improved image quality with CLAHE and continued with contour smoothing and removal of thin objects using morphology opening. The next process is to improve the image contrast with CLAHE so that the blood vessel features in the retinal image are seen more clearly. The image resulting from CLAHE is subjected to an image patching process by dividing the image into small pieces of 5.000 patches per image. In the image patching process, because the training data used are 20 images, the resulting patch is 100.000 patches with a size of 64×64 pixel.

3.2 BVU-Net application

At this stage, architectural changes will be made, with the U-Net architecture serving as the foundation. The VGG architecture is combined on the encoder part, and batch normalization is added throughout each convolution operation to modify the architecture. Batch normalization is used to equalize activation in each layer to accelerate training. Fig. 4 shows the altered architecture. The modified architecture BVU-Net is consisted of two parts, namely the left side (encoder) and the right side (decoder). The encoder path uses 5 convolutional blocks consisting of convolutional layer, ReLU, batch normalization, and max pooling. The initial step taken in the first block in the encoder path is that the pre-processed image is used as an input image with a size of 64×64 pixels. This is the first block on the encoder path. The image then enters the green boxed convolutional layer process. A 3×3 convolution process and a ReLU activation function process. Additionally, a blue box signifies the addition of batch normalization. The next step is to reduce the size of the feature map using a max-pooling size of 2×2. Repeat the convolutional layer process and max pooling with 128 kernels in the second block. The third to fifth blocks repeat the previous block's process, with the exception that the convolutional layer process is repeated twice. The third block has 256 kernels, the fourth block has 512, and the fifth block has 512. A convolutional layer process with 512 kernels is carried out in the connecting section between the two architectures.

Convolutional layer, ReLU, and up-sampling are three of the five convolutional blocks in VGG that are used in the decoder path. Convolution results on the connecting path are obtained by increasing the dimensions of the feature map size using up-sampling measuring 2×2 and the number of kernels up to 512. Furthermore, the convolution results of the fifth encoder path block are combined with the convolution results of the first decoder path block, yielding a total of 768 kernels. The merging results continue with 2×2 up-sampling after entering the convolutional layer process as in the encoder path. The same procedure is performed in blocks two through five, where there are as many as 768 kernels in block two, 384 in block three, 192 in block four, and 96 in block five. The last step is to carry out the convolution process with 1 kernel to reshape the segmented image and do along with the Sigmoid activation function.

In the modified architecture, the convolution process is carried out with Eqs. (7), (8), and (9). For example, to get the convolution value on the 1st feature maps, the 1st input (A_1^1) is to perform a

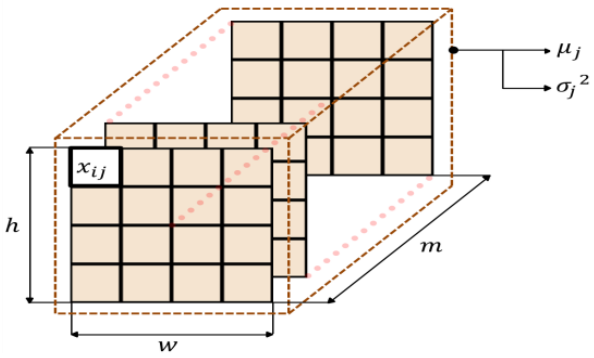


Figure. 6 Batch normalization illustration

convolution operation between the 1st image input (C_1) resulting from the addition of the same padding (adjusting the image size with the kernel) and the 1st kernel(K_1). Convolution operation results are added to bias 1 (b_1). The obtained values are substituted into the ReLU activation function using Eq. (10), and each activation layer is normalized using batch normalization. Fig. 6 shows an illustration of the batch normalization process. Fig. 6 shows several feature maps, including height (h) and width (w) feature maps. Each feature map has a pixel value (x_{ij}), which is the input matrix entry in the i-th row and j-th column. Determine the number of mini-batches (m) for each feature map. Then, using Eqs. (11) and (12), calculate the mean (μ_j) and variance (σ_j^2) for each mini-batch. Each subsequent matrix entry is normalized using Eq. (13) to generate a new matrix entry value $(\hat{x}_{ij})$. The resulting value will be in a value range that is not too far between the highest and lowest values.

3.3 Training data

The data training process is carried out using a modified BVU-Net architecture, resulting in a BVU-Net model. The model is trained with 100,000 training data which is divided into two parts so that 60,000 data is obtained for the trained data and 40,000 for the data tested in the training process. At this stage, the number of epochs is 30, and the number of samples in one iteration (batch size) is 128. In Fig. 7 and 8, it can be seen that the accuracy of the training data (blue line) and the accuracy of the validation data or validation accuracy (orange line) in the DRIVE and STARE datasets always increases at each epoch. In the first epoch of the DRIVE dataset, the accuracy obtained is 58% and the validation accuracy is 41%. The accuracy curve gradually rises to 0.98 in the following epoch, while the validation accuracy rises to 0.94. Meanwhile, in the first epoch of the STARE dataset, the accuracy obtained 21% and the validation accuracy is 19%. The accuracy increases steadily towards 99% in the following epoch, while the validation accuracy also increases to 96%. The graph shows that the accuracy of the training and validation data in both datasets is not overfitting.

The loss graph obtained during the training process is provided, as shown in Figs. 9 and 10. In Figs. 9 and 10, it can be seen that the loss function on the training data (blue line) and the loss function value in the validation data (orange line obtained during the training process in the DRIVE and STARE datasets always decreases. The loss graphs in Figs. 9 and 10 show no overfitting because the loss curves in the training data and testing data have the same pattern. In addition, the gap between training loss and validation loss is not far. The graphs of accuracy and loss in both the DRIVE and STARE datasets have good patterns and there is no overfitting. This means that the model can work very well not only on training data but also on data that has never been trained before. From the gaps, it can be concluded that the architecture works excellent because the accuracy obtained is more than 98% and the loss obtained is close to 0. The last step of the training is to store the best weights for testing data.

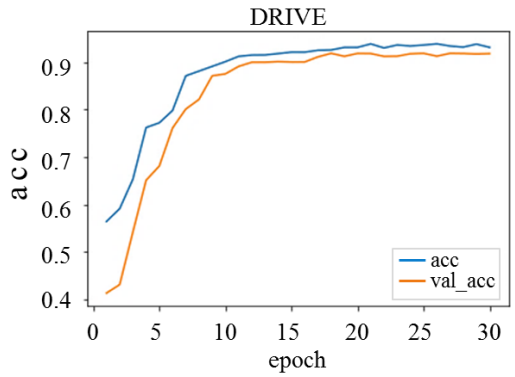


Figure. 7 The accuracy result on training data and validation data in training process on DRIVE dataset

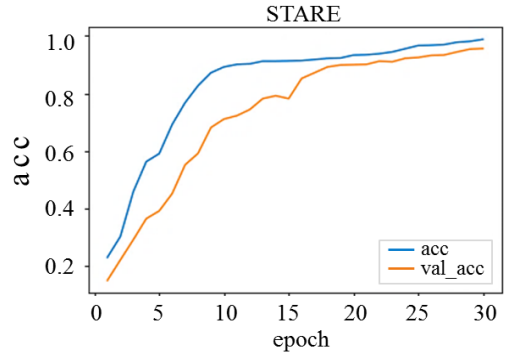


Figure. 8 The accuracy result on training data and validation data in training process on STARE dataset

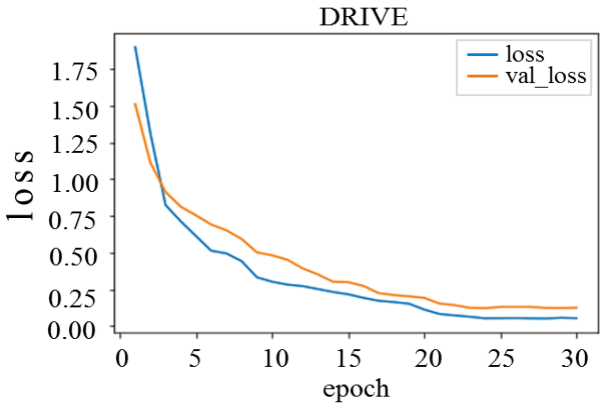


Figure. 9 The loss result on training data and validation data in training process on DRIVE dataset

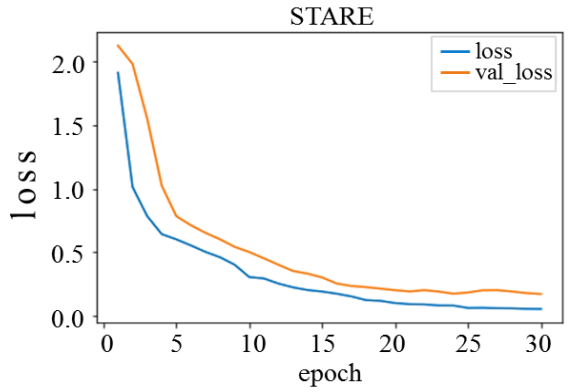


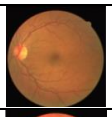
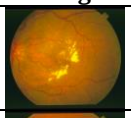
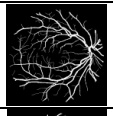
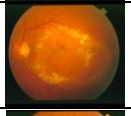
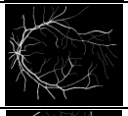
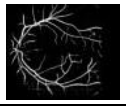
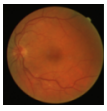
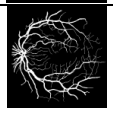
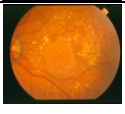
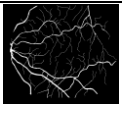
Figure. 10 The loss result on training data and validation data in training process on STARE dataset

3.4 Testing

In the testing process, the testing image is also patched in small patches sequentially. After the patching process, the weights obtained from the training process are applied to the patched image to segment the retinal blood vessels in the image. The results of the patched images that have been segmented are merged into a segmented image that has the same size as the original size. The Testing process is carried out to validate the training model. Table 1 shows the comparison of the blood vessel of retinal image segmentation results from the proposed method with the ground truth available in the dataset. In some images, the results of retinal blood vessel segmentation are slightly different from the ground truth, especially for thin blood vessels. Some thin blood vessels are not detected in the segmented image. It can be shown in Fig. 11. In Fig. 11, the yellow box is an example of a subset of the image in the DRIVE dataset which shows detectable blood vessels and undetectable blood vessels by segmented images using BVU-Net. In Fig. 11. The blue box is also an example of the image section in the STARE dataset which shows the blood vessels that have been successfully and unsuccessfully detected by BVU-Net. In Fig. 11, it can be seen that the blood vessels of the segmented image on the STARE dataset are more clearly visible than the segmented image on the DRIVE. However, the results of segmentation on both datasets also show that some thin blood vessels can be detected.

The ROC shows the relationship between false positive rate (FPR) and true positive rate (TPR by calculating the area under curve (AUC) on the ROC curve. The larger the UAC area, the better the model or architecture in recognizing the label of each data In the DRIVE dataset, the AUC area of 0.975 is obtained, while the STARE dataset is 0.8849. Based on the AUC value, it shows that the results of the model performance for retinal blood vessel segmentation in both datasets have good quality. In Figs. 12 and 13, it can be seen that the ROC curve graph illustrates the relationship between false positive rate (FPR) and true positive rate (TPR), where the ROC curve is carried out by calculating the

Table 1. The results of retinal blood vessel segmentation in the drive and stare; original, ground truth and segmented image results of BVU-Net

No	DRIVE			STARE		
	Original Image	Ground Truth	Result	Original Image	Ground Truth	Result
1						
2						
3						

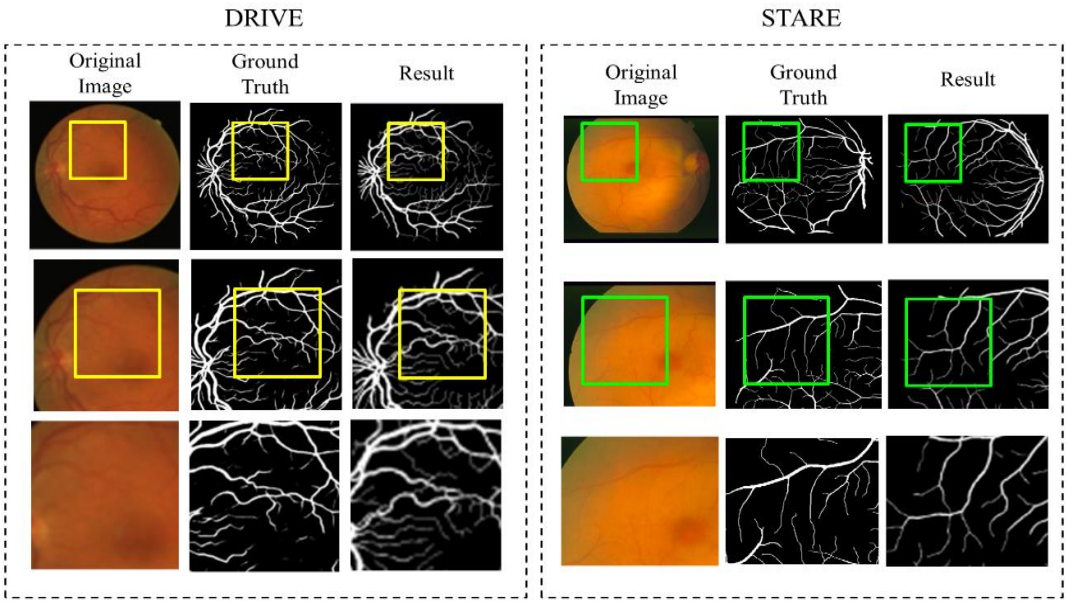


Figure. 11 The Comparison of Retinal Blood Vessel Segmentation Results on Proposed Method and Ground Truth

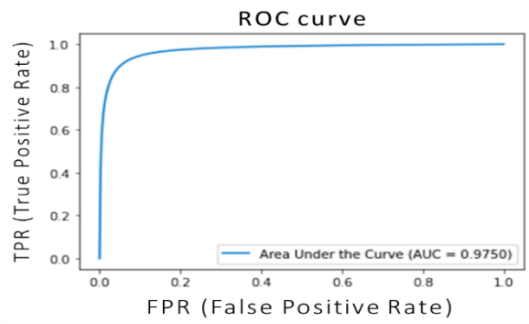


Figure. 12 Graph of ROC curve obtained during the testing process on the DRIVE dataset

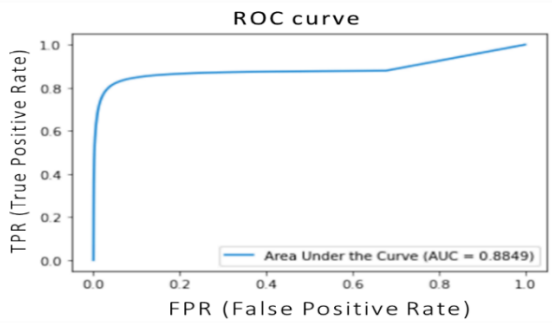


Figure. 13 Graph of ROC curve obtained during the testing process on the STARE dataset

area under curve (AUC) on the ROC curve. In the DRIVE dataset, the AUC area of 0.975 is obtained, while the STARE dataset is 0.8849. Based on the AUC value, it shows that the results of the model performance for retinal blood vessel segmentation in both datasets have good quality.

Table 2. The comparison of performance results in DRIVE daraset of proposed method with other studies

Method	Acc (%)	Sen (%)	Spe (%)	F1 (%)	IoU
Convolutional Autoencoder [22]	96	73	92	-	-
Convolutional Neural Network[21]	94.9	78.2	97.6	80.3	0.67
MSFFU-Net [23]	96.9	77.62	98.3	-	-
UNet with Genethic Algorithm[20]	95	75.06	98.5	80.8	-
Stride UNet[10]	94.8	73.9	95.6	-	-
Proposed method	96.39	78.71	98.1	78.9	0.65

Table 3. The comparison of performance results in STARE daraset of proposed method with other studies

Method	Acc (%)	Sen (%)	Spe (%)	F1 (%)	IoU
Stride UNet [10]	94.7	74.8	96.2	-	-
DenseUNet [24]	96.51	68.07	99.16	-	0.77
Residual Block Incorporated [25]	95.37	55.82	98.62	64.68	0.48
EffUnet [26]	95.69	75.54	99.7	-	-
Improved UNet [27]	96.83	63.29	99.67	80.49	0.67
Proposed method	96.83	79.35	98.49	81.29	0.69

3.5 Analysis and interpretation of results

This study has offered a new BVU-Net architecture for blood vessel segmentation on retinal images. To declare the success of the BVU-Net architecture so the results of the BVU-Net performance were compared with the results of the CNN architecture performances in other similar studies. A comparison of the performance results of BVU-Net with other architectures CNN can be seen in Table 2 and Table 3. Table 2 shows the results of the comparison of the performance of each architecture on DRIVE dataset, while Table 3 is on STARE dataset. The performance results that are used as comparisons in Table 2 and Table 3 are accuracy, sensitivity, specificity, F1-score and IoU. The performance results of the BVU-Net architecture on the DRIVE dataset (Table 2) shows the results of accuracy, sensitivity and specificity are better than other studies results. Unfortunately, The highest F1-score was obtained by [20] using U-Net that has been modified with Genetic algorithm. The highest IoU is obtained by [21] using conventional U-Net. The F1-score and IoU results of BVU-Net are not the best, but the F1-score is close to 1 and IoU is above 0.5. This means that BVU-Net has good precision and recall in recognizing each class and is good in overcoming overlapping in each class (foreground and background) in the DRIVE dataset.

Table 3 shows the performance results studies on the STARE dataset. The accuracy, sensitivity and F1-score of this study in STARE dataset obtained the highest results compared to other studies. For specificity and IoU, the best specificity and IoU results were obtained by [24]. The specifications in this study are lower than some other studies, but the sensitivity value obtained in this study is the highest compared to other studies. Sensitivity and specificity provide opposite measures, where the higher the sensitivity, the lower the specificity or vice versa. The higher sensitivity indicates the proposed method is able to recognize the positive class better than other studies. However, the IoU obtained in this study was above 0.5, indicating that the BVU-Net architecture is good in overcoming the overlap area between classes.

From the performance comparison in Table 2 and Table 3, it can be seen that BVU-Net has highest sensitivity that other studies both on DRIVE and STARE. Other studies did not measure F1-scores and IoU. It cannot be concluded whether the architecture used has a balanced precision and recall and is able to overcome overlapping. It can be concluded that BVU-Net is robust to detect blood vessels in retinal images well, but it is still necessary to develop the BVU-Net architecture to be able to improve the ability to overcome overlapping areas between blood vessels (foreground) and back ground.

4. Conclusions

Based on the results of the research and discussion, it can be concluded that the results of the model performance on the BVU-Net architecture in the retinal blood vessel segmentation is robust in both datasets, where the average accuracy is above 96%, specificity is above 98%. Sensitivity and F1-score are above 78%, but IoU is still below 70%. These results indicate that the ability of model to overcome overlapping between background and foreground on retinal blood vessel segmentation so it is sometimes thin blood vessel cannot be detected. The architecture should be improved to increase the F1-score and IoU to get more significant and accurate results for blood vessels segmentation on retinal images.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

We certify that all authors contributed to this study. Paper conceptualization and methodology, Anita Desiani and Erwin; software and validation, Yogi Wahyudi, Muhammad Arhami and Bambang Suprihatin; formal analysis and investigation, Erwin and Endro Setyo Cahyono; resources, data curation and writing-original draft preparation, Anita Desiani, Yogi Wahyudi and Bambang Suprihatin; writing-review editing and visualization, Erwin, Endro Setyo Cahyono and Muhammad Arhami; supervision and project administration. Erwin, Bambang Suprihatin and Anita Desiani.

Acknowledgments

The authors thank to computation laboratory of mathematics and natural science faculty, universitas sriwijaya for all the support on our research.

References

[1] Maison, T. Lestari, and A. Luthfi, “Retinal Blood Vessel Segmentation using Gaussian Filter”, *Journal of Physics: Conference Series*, Vol. 1376, No. 1, Nov. 2019, doi: 10.1088/1742-6596/1376/1/012023.

[2] P. M. Samuel and T. Veeramalai, “Review on Retinal Blood Vessel Segmentation - An

Algorithmic Perspective", *Int. J. Biomed. Eng. Technol.*, Vol. 34, No. 1, pp. 75–105, 2020, doi: 10.1504/IJBET.2020.110362.

[3] M. Monteiro and A. Sarmento, "A HW/SW System to Detect Druses in Retinal Fundus Image for The Diagnostic of Age Related Macular Degeneration", In: *Proc. of IEEE Symposium on Computer-Based Medical Systems*, Vol. 2020-July, pp. 297–302, Jul. 2020, doi: 10.1109/CBMS49503.2020.00063.

[4] A. Ali, W. M. D. W. Zaki, and A. Hussain, "Retinal Blood Vessel Segmentation from Retinal Image using B-COSFIRE and Adaptive Thresholding", *Indones. J. Electr. Eng. Comput. Sci.*, Vol. 13, No. 3, pp. 1199–1207, 2019, doi: 10.11591/ijeecs.v13.i3.pp1199-1207.

[5] A. Desiani, Erwin, B. Suprihatin, S. Yahdin, A. I. Putri, and F. R. Husein, "Bi-path architecture of CNN segmentation and classification method for cervical cancer fisorders based on pap-smear images", *IAENG Int. J. Comput. Sci.*, Vol. 48, No. 3, 2021.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation olaf", *Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Vol. 9351, pp. 234–241, 2015, doi: 10.1007/978-3-319-24574-4_28.

[7] A. A. Pravitasari, N. Iriawan, M. Almuhayar, T. Azmi, Irhamah, K. Fithriasari, S. W. Purnami, and W. Ferriastut, "UNet-VGG16 with transfer learning for MRI-based brain tumor segmentation", *Telkomnika (Telecommunication Comput. Electron. Control.*, Vol. 18, No. 3, pp. 1310–1318, 2020, doi: 10.12928/TELKOMNIKA.v18i3.14753.

[8] M. Melinsca, P. Prentasic, and S. Loncaric, "Retinal Vessel Segmentation using Deep Neural Networks", In: *Proc. of VISAPP 2015 - 10th Int. Conf. Comput. Vis. Theory Appl. VISIGRAPP*, Vol. 1, pp. 577–582, 2015, doi: 10.5220/0005313005770582.

[9] H. Fu, Y. Xu, D. W. K. Wong, and J. L. Ocular, "Retinal Vessel Segmentation Via Deep Learning Network And Fully-Connected Conditional Random Fields", In: *Proc. of 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, pp. 698–701, 2016.

[10] T. A. Soomro, O. Hellwich, A. J. Afifi, M. Paul, J. Gao, and L. Zheng, "Strided U-Net Model: Retinal Vessels Segmentation using Dice Loss", In: *Proc. of 2018 International Conference on Digital Image Computing: Techniques and Applications, DICTA 2018*, pp. 1–8, 2018, doi: 10.1109/DICTA.2018.8615770.

[11] S. Pasban, S. Mohamadzadeh, J. Z. Moghaddam, and A. K. Shafiei, "Infant brain segmentation based on a combination of vgg-16 and u-net deep networks", *IET Image Process.*, Vol. 14, No. 17, pp. 4756–4765, 2020, doi: 10.1049/iet-ipr.2020.0469.

[12] Y. Mei, H. Jin, B. Yu, E. Wu, and K. Yang, "Visual geometry Group-UNet: Deep learning ultrasonic image reconstruction for curved parts", *J. Acoust. Soc. Am.*, Vol. 149, No. 5, pp. 2997–3009, 2021, doi: 10.1121/10.0004827.

[13] T. S. Sharan, S. Tripathi, S. Sharma, and N. Sharma, "Encoder Modified U-Net and Feature Pyramid Network for Multi-class Segmentation of Cardiac Magnetic Resonance Images", *IETE Tech. Rev.*, Vol. 0, No. 0, pp. 1–13, 2021, doi: 10.1080/02564602.2021.1955760.

[14] N. Wattanavichean, J. Boonchai, S. Yodthong, C. Preuksakarn, S. C. H. Huang, and T. Surasak, "Gfp pattern recognition in raman spectra by modified vgg networks for localisation tracking in living cells", *Eng. J.*, Vol. 25, No. 2, pp. 151–160, 2021, doi: 10.4186/ej.2021.25.2.151.

[15] M. Liu, W. Wu, Z. Gu, Z. Yu, F. F. Qi, and Y. Li, "Deep learning based on Batch Normalization for P300 signal detection", *Neurocomputing*, Vol. 275, pp. 288–297, 2018, doi: 10.1016/j.neucom.2017.08.039.

[16] N. M. Aboelenein, P. Songhao, A. Koubaa, A. Noor, and A. Afifi, "HTTU-Net: Hybrid Two Track U-Net for Automatic Brain Tumor Segmentation", *IEEE Access*, Vol. 8, pp. 101406–101415, 2020, doi: 10.1109/ACCESS.2020.2998601.

[17] L. Liu, L. Mou, X. X. Zhu, and M. Mandal, "Skin Lesion Segmentation Based on Improved U-net", In: *Proc. of 2019 IEEE Can. Conf. Electr. Comput. Eng. CCECE 2019*, pp. 1–4, 2019, doi: 10.1109/CCECE.2019.8861848.

[18] S. H. Majeed and N. A. M. Isa, "Iterated Adaptive Entropy-Clip Limit Histogram Equalization for Poor Contrast Images", *IEEE Access*, Vol. 8, pp. 144218–144245, 2020, doi: 10.1109/ACCESS.2020.3014453.

[19] A. Desiani, Erwin, S. I. Maiyanti, B. Suprihatin, N. Rachmatullah, A. N. Fauza, and I. Ramayanti, "R-Peak detection of beat segmentation and convolution neural network for Arrhythmia classification", *J. Eng. Sci. Technol.*, Vol. 17, No. 2, pp. 1231–1246, 2022.

[20] T. Laibacher, T. Weyde, and S. Jalali, "M2U-net: Effective and efficient retinal vessel segmentation for real-world applications", *IEEE*

Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work., Vol. 2019-June, pp. 115–124, 2019, doi: 10.1109/CVPRW.2019.00020.

[21] N. Brancati, M. Frucci, D. Gragnaniello, and D. Riccio, "Retinal Vessels Segmentation Based on a Convolutional Neural Network", *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, Vol. 2, pp. 119–126, 2018, doi: 10.1007/978-3-319-75193-1.

[22] A. Ortiz, J. Ramírez, R. C. Arándiga, M. J. G. Tarifa, F. J. M. Murcia, and J. M. Górriz, "Retinal Blood Vessel Segmentation by Multi-channel Deep Convolutional Autoencoder", *Adv. Intell. Syst. Comput.*, vol. 771, pp. 37–46, 2019, doi: 10.1007/978-3-319-94120-2_4.

[23] D. Yang, G. Liu, M. Ren, B. Xu, and J. Wang, "A Multi-Scale Feature Fusion Method based on U-Net for Retinal Vessel Segmentation", *Entropy*, Vol. 22, No. 8, 2020, doi: 10.3390/E22080811.

[24] X. Li, H. Chen, X. Qi, Q. Dou, C. W. Fu, and P. A. Heng, "H-DenseUNet: Hybrid Densely Connected UNet for Liver and Tumor Segmentation from CT Volumes", *IEEE Trans. Med. Imaging*, Vol. 37, No. 12, pp. 2663–2674, 2018, doi: 10.1109/TMI.2018.2845918.

[25] T. Mostafiz, I. Jarin, S. A. Fattah, and C. Shahnaz, "Retinal Blood Vessel Segmentation Using Residual Block Incorporated U-Net Architecture and Fuzzy Inference System", In: *Proc. of IEEE International WIE Conference on Electrical and Computer Engineering, WIECON-ECE 2018*, pp. 106–109, 2018, doi: 10.1109/WIECON-ECE.2018.8783182.

[26] M. R. Mathews, S. M. Anzar, R. Kalesh Krishnan, and A. Panthakkan, "EfficientNet for retinal blood vessel segmentation", In: *Proc. of 2020 3rd International Conference on Signal Processing and Information Security, ICSPIS 2020*, pp. 4–7, 2020, doi: 10.1109/ICSPIS51252.2020.9340135.

[27] Z. Huang, Y. Fang, H. Huang, X. Xu, J. Wang, and X. Lai, "Automatic Retinal Vessel Segmentation Based on an Improved U-Net Approach", *Sci. Program.*, Vol. 2021, 2021, doi: 10.1155/2021/5520407.