Received: July 8, 2022. Revised: September 26, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A Pyramid Model of Convolutional Neural Network to Classify Acute Lymphoblastic Leukemia Images

Arif Muntasa[1*] Rima Tri Wahyuningrum[1] Zabrina Tuzzahra[2] Abdelwahed Motwakel [3] Muhammad Yusuf[1] Wayan Firdaus Mahmudi[4]

[1] *Universitas Trunojoyo Madura, East Java, Indonesia*
[2]*UPN Veteran, East Java, Indonesia*
[3]*Department of Computer and Self Development, Preparatory Year Deanship, Prince Sattam bin Abdulaziz University, AlKharj, Saudi Arabia*
[4]*Universitas Brawijaya, Malang, East Java, Indonesia*
*Corresponding Author's Email: arifmuntasa@trunojoyo.ac.id

Abstract: Many researchers have classified acute lymphoblastic leukemia using several methods. One of the methods is a convolutional neural network. However, the limitation of the convolutional neural network is a large number of trainable parameters updated. The paper proposes a new architecture based on the convolutional neural network. We have designed and implemented a convolutional neural network with different kernels, where we increase the number of kernels like pyramid models. We utilized the final convolution to conduct the fully connected, followed by the SoftMax function to classify the image. We have evaluated our proposed architecture using acute lymphoblastic leukemia image database (ALL-IDB2). The results show that our proposed method produced the average *Accuracy*, *Precision*, and *Recall* of 99.17%, 99.33%, and 99%, respectively. It has outperformed other models, i.e., shape features, Multi distance of GLCM, CNN and SVM, Pretrained deep convolutional neural networks, AlexNet, ensemble network, convolutional and recurrent neural network, and hypercomplex-valued convolutional neural networks.

Keywords: Leukemia classification, Convolutional, Neural network, A new architecture.

1. Introduction

Leukemia is one of the most dangerous and feared cancers. Therefore, we should always conduct early detection to avoid fatal effects. If the body produces more abnormal white blood cells, bone narrowing will be disturbed, and if it has occurred, the white blood cells cannot work effectively. Leukemia has a similar indication to other diseases. Unfortunately, it needs a high cost to check the white blood cell to detect Leukemia disease. This paper aims to classify acute lymphoblastic leukemia by our new proposed architecture. Leukemia image classification is one of the ways to detect Leukemia disease where Leukemia classification will separate an image into two parts, i.e., positive or negative Leukemia [1]. Many researchers have conducted Leukemia image classification. [2] has proposed Leukemia image classification using three stages, i.e., pre-processing, feature extraction, and classification. They employed cyan, magenta, yellow, and key (CMYK) color models to separate white blood cells (WBC). They also implemented histogram equalization to identify the color image distribution. However, the process is hard to be segmented between the WBC and image background. The segmentation error will affect the feature extraction and classification results. Therefore, the method is not an efficient model for classifying Leukemia.

Similarly, the color-based segmentation method has been proposed to classify acute lymphoblastic leukemia using main stages, pre-processing, segmentation, feature extraction, and classification [3]. They have employed the shape and texture feature to represent the object of the image. The results show that 95.38% accuracy has been obtained.

The accuracy depends on the segmentation results, and image segmentation errors could cause misclassification. The weakness of the paper has been improved by image segmentation enhancement and followed by texture features generated, i.e., homogeneity, entropy, energy, and contrast. Four features can increase the accuracy to 96.67% [4]. It shows that the improvement of the image segmentation process only increases 1.29% accuracy. The investigation results show that the segmentation improvement can not guarantee a significant increase in accuracy.

Other researchers have built deep learning architecture for Leukemia classification and conducted data augmentation to increase training set variation. They have also tried to avoid over-fitting by transfer learning using dense convolutional neural network (DCNN) architecture and ensembled technique to obtain the feature extraction results. Some DCNN has been implemented to classify Leukemia, i.e., AlexNet, VGGNet-16, VGGNet19, MobileNet, ShuffleNet, and two NASNet, InceptionV3, Xception, DenseNet20, and MobileNet. They assumed that increasing the datasets for the training process could produce a better model. They have made 96.58% accuracy [2]. The limitation of the method is that the model takes longer to build; therefore, the model needs a high cost to classify the images.

Furthermore, acute lymphoblastic leukemia classification is proposed using dense convolutional neural network (DCNN) [5]. They also carried out the augmentation of the datasets. They selected the features using univariate feature selection before classification. They employed acute lymphoblastic Leukemia–image database 1 (ALL-IDB1) as the dataset to evaluate their proposed method. They randomly separated 75% and 25% as the training and testing sets. The experimental results produced 97.2% accuracy.

In addition, the visual geometry group has proposed VGG16 and VGG19 architectures to classify the Leukemia image from the C-NMC 2019 dataset [6]. They also employed xception convolutional neural network to extract and classify the datasets. They initialized 0.00001 as a learning rate with 400 epochs. The results show that accuracy for VGG16, VGG19, and Xception are 92.48%, 91.59%, and 90.41%, respectively. While F1 scores for the proposed method are 92.60%, 91.75%, and 90.76% for VGG16, VGG19, and exception, respectively. However, they are new architectures with bulky trainable parameters. Therefore, these architectures required high-performance devices and took longer to extract and classify the images.

A new approach of transfer learning using VGG-f to extract the features is proposed to improve the VGG16 and VGG19. They utilized a support vector machine to classify the feature extraction results. VGG-f is a convolutional neural network architecture that is built based on AlexNet. The VGG-f employed smaller kernels for the convolution process in the first, third, and fourth [7]. However, they still have 4096 neurons for fully-connected layers like AlexNet architecture. They improved the image by converting the actual color into CIELAB color space. Image segmentation is conducted to separate the main object and background, and the results were utilized to calculate the shape features. Finally, the support vector machine is used to classify the image. They claimed that the proposed method had been evaluated using large databases, i. e., ALL-IDB1. ALL-IDB2, Leukocytes, and CellaVision datasets. The results show that the VGG-f has produced 99% accuracy.

Leukemia image classification was also conducted using contrast enhancement to improve the image quality and morphological color segmentation to separate the primary object and the background. In addition, fuzzy C-mean is employed to classify the features found [8]. They used acute lymphoblastic leukemia image database 1 (ALL-IDB1) to evaluate the method. The results show that the accuracy is 98%. However, an evaluation of the proposed architecture must be conducted using the different indexes so that the proposed method can prove its reliability.

Knowledge-based morphology was proposed and combined with CD markers to classify Leukemia images using morphology and cluster of differentiation (CD) markers to decide the results. They extract the blood white cell using four parameters, i.e., number of nucleoli, color, texture, and shape. The extraction results are processed using CD markers classification. The proposed algorithm has produced 99.67% accuracy. Unfortunately, they do not compute a computation time to classify the images [9].

A different method for image segmentation of Acute lymphoblastic Leukemia was proposed to separate the main object and background by using fuzzy. Furthermore, they employed a support vector machine (SVM) to classify acute lymphoblastic Leukemia, and their proposed method produced 94.73% average accuracy. However, the classification results are still under ninety percent. Therefore, the plans still need to be improved, especially the feature extraction and classification parts [10].

Currently, the trending method has been

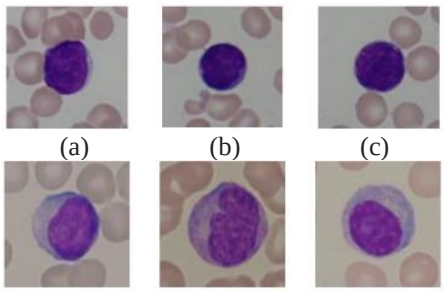

Figure. 1 Image samples of acute lymphoblastic Leukemia: (a), (b), and (c) are Positive Leukemia, (d), (e), and (f) are Healthy Images.

developed by many researchers, i.e., convolutional neural network (CNN). However, CNN also has a weakness, where CNN takes longer time than a classical neural network. Some papers have proposed a convolutional neural network to classify acute lymphoblastic Leukemia [11, 12]. They offered a pre-trained deep neural network to organize the images, i.e., AlexNet architecture, and they modified the datasets using the augmentation technique to reduce overtraining. They employed NVIDIA GeForce GTX 960M DDR5 4096 MB GPU to process the training and testing. However, their proposed method has produced 99.5% and 96.06% accuracies without image segmentation. Unfortunately, they need around thirty-five minutes to train the acute lymphoblastic Leukemia images. The AlexNet architecture has updated 34.944, 614.656, 885.120, 1.327.488, and 884.992 trainable parameters on the feature extraction layer, i.e., for the first, second, third, fourth, and fifth, respectively. The architecture also updated three Fully-connected layers, which are 37.752.832, 16.781.312, and 4.097.000 trainable parameters.

Residual network (ResNet) is one of the convolutional neural network architectures that has many variants, i.e., ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-110, ResNet-152, ResNet-164, and ResNet-1202. The number behind the ResNet indicates several layers used. However, the simplest ResNet architecture has eighteen layers dan the most complex architecture consists of a thousand and two hundred-two layers. The more layers used, the longer time required. Therefore, it needs a high-specification machine to solve a task in computer vision [13, 14].

Visual geometry group (VGG) also has two models, i.e., VGG-16 and VGG-19. VGG-16 consists of thirteen convolution layers and three fully connected layers, while VGG-19 has sixteen convolution and three fully connected layers, improving the results of VGG-16. However, VGG-16 ad VGG-19 are complex architectures for solving computer vision tasks. Therefore VGG-16 or VGG-19 takes longer time than AlexNet [15, 16].

Based on the above explanations, we proposed a new convolutional neural network architecture to classify the acute lymphoblastic Leukemia images. The number of kernels increased from the first until the last convolution, like a pyramid. Therefore, our proposed architecture is called as pyramid model of convolutional neural network (PM-CNN). This architecture is like a pyramid, which has gradually employed kernels. Several kernels are carried out following a multiple of the initial value. We carried out six convolutional followed by normalization, ReLU activation functions, and max-pooling to reduce the feature map in the feature extraction layer. Therefore, our proposed architecture consists of twenty-four layers in the feature extraction layer. We apply five kernels for the first convolution. Secondly, we increase the number of kernels twice the previous. A similar process is conducted until the last convolution.

2. Proposed method

We employed two hundred and sixty images to evaluate our proposed architecture, where the images are separated into two classes, i.e., a hundred and thirty healthy images and the rest images are Leukemia. Some images are hard to be distinguished between healthy and Leukemia images. Therefore, the role of the machine is needed to classify the image quickly. Here are the image samples of the acute lymphoblastic leukemia image database (ALL-IDB2). Images are taken using microscopic, where the blood is dripped with a substance so that when captured using a tool, the color changes to purple, as seen in Fig. 1. The original image resolution is 257 rows and 257 columns, where the image has three channels.

We proposed a new convolutional neural network architecture. Our proposed method has an input, feature extraction, and classification layer. We use acute lymphoblastic Leukemia (ALL) as image input, where we employ the original image size: 257 heights, 257 widths, and three channels. We occupy four activities in the feature extraction layer: convolutions, batch normalization, activation function, and pooling. We proposed six convolution, six batch normalization layers, six activation function layers, and six pooling layers in the feature extraction layer. Therefore, we have 24 layers in feature extraction layers. In the classification layer, we proposed three layers, i.e., fully connected, an activation function, and classification layers, as shown in Fig. 2 and Table 1. In the feature extraction layer, we have employed six convolution processes, where we occupy various

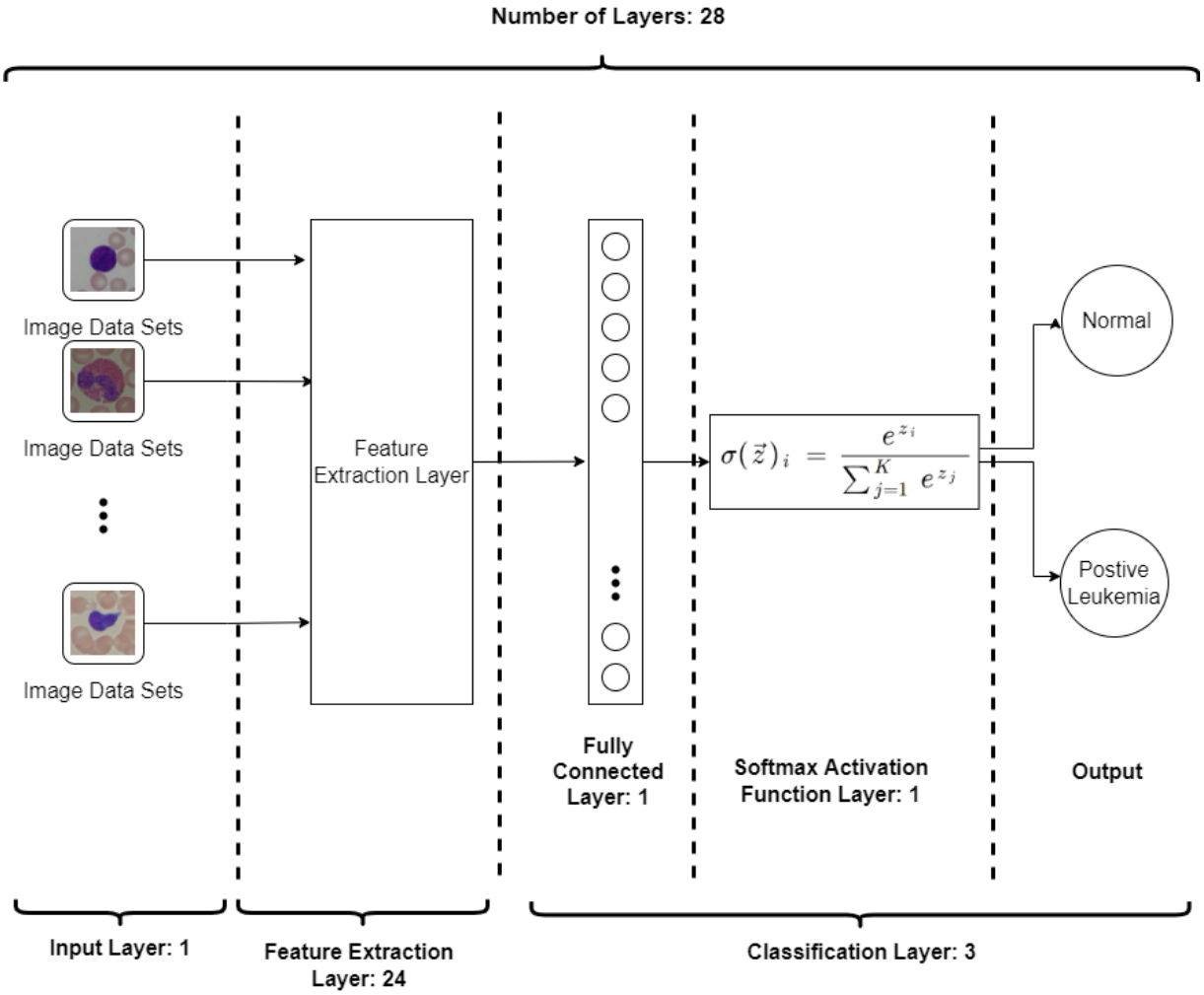

Figure. 2 Our proposed method: new architecture of convolutional neural network

Table 1. Number of layers proposed method

No	Layer	Description	Number of Layers
1	Input	Image 257 x 257	1
2	Feature extraction	Convolution	6
		Batch normalization	6
		Activation function	6
		Pooling	6
3	Classification	Fully connected	1
		Activation function	1
		output	1
			28

numbers of kernels (D'') in sequence, namely are D'', $2D''$, $3D''$, $4D''$, $5D''$, and $6D''$, where $D'' \in \{5\}$, therefore the number of kernels is 5, 10, 15, 20, 25, and 30. In this case, we employ five different kernel sizes, which is $K_w \in \{3, 5, 7, 9, 11\}$. In general, we build layer models of our convolutional neural network architecture, as shown in Fig. 3.

Fig. 4 shows that the first convolution process uses five kernels, whose size is 3x3. We apply stride [1 1], where the kernel matrix will move one column and one row. The convolution result produces the same size as the original input. We have considered our CNN architecture's stride to avoid much information loss. The more significant the stride size, the more information loss. If the stride size is smaller than kernel size, many elements overlap during the convolution process. The more image pixels that overlap during the convolution process, the better the features formed.

If $\mathcal{X}$ variable represents the input image and $\mathcal{X} \in \mathbb{R}^{H \times W \times D}$, then the symbols of $\mathbb{R}, H, W,$ and D represent the space, height, width, and channel of the image. If $\mathcal{X} \in \mathbb{R}^{H \times W \times D}$ is convoluted by multidimensional kernel matrix $\mathcal{F} \in \mathbb{R}^{H' \times W' \times D \times D''}$. Symbols of H', W', and D'' describe kernel height, width, and the number of kernels, respectively. The convolutional results between $\mathcal{X} \in \mathbb{R}^{H \times W \times D}$ image

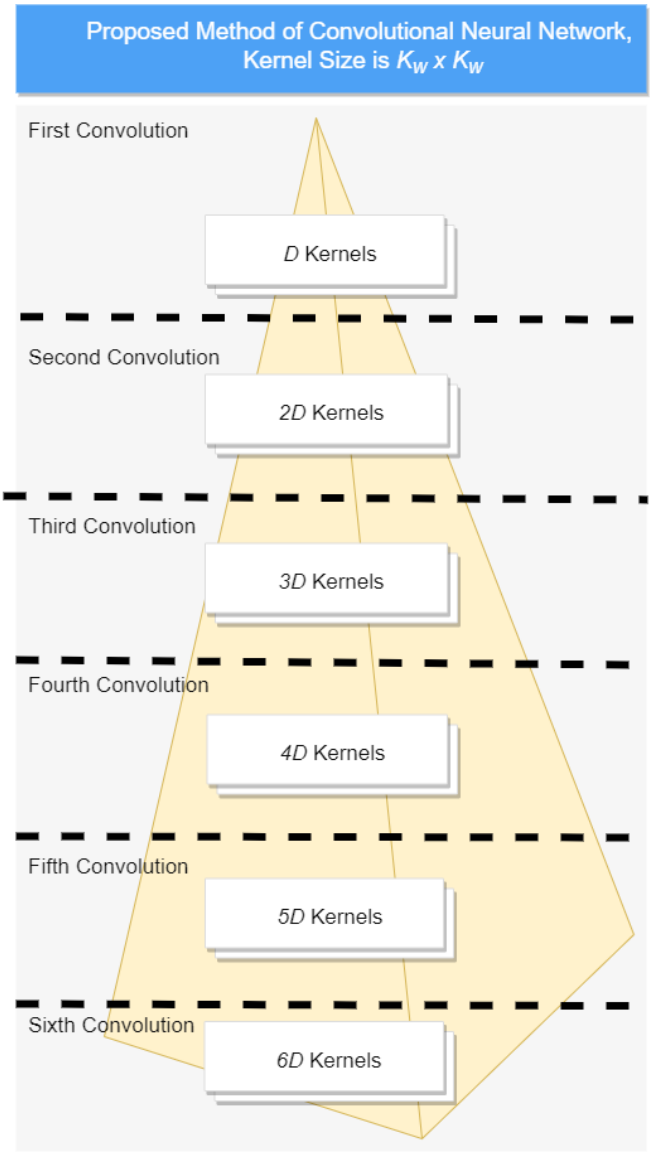

Figure. 3 Our proposed model: pyramid model of convolutional neural network

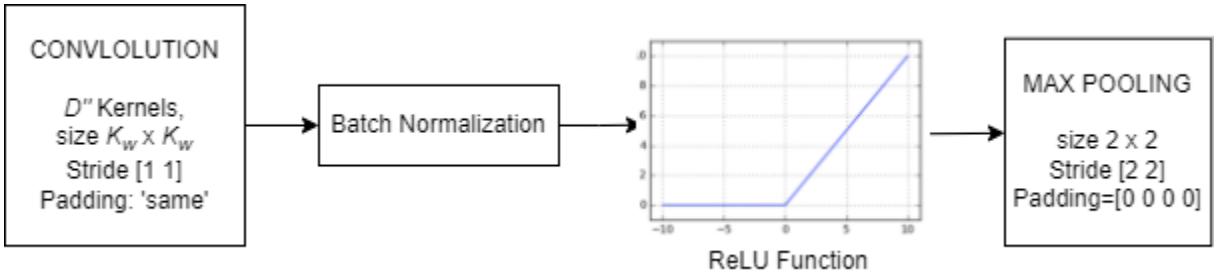

Figure. 4 Our proposed method: the first convolution, batch normalization, ReLu, and max pooling

and $\mathcal{F} \in \mathbb{R}^{H' \times W' \times D \times D''}$ can be expressed by using Eq. (1)

$$\mathcal{Y}_{i'' \, j'' \, d''} = \sum_{i'=1}^{H'} \sum_{j'=1}^{W'} \sum_{d'=1}^{D} \mathcal{F}_{i^i \, j^i \, d} \times \mathcal{X}_{i''+i'-1, j''+j'-1, d', d''} \quad (1)$$

In Eq. (1) $i', j',$ and d' represent an index of a row, a column, and a channel of an image, whereas $i'', j'',$ and d'' describe the new row, column, and channel of the image pixel results. The image convolutional result as written in Eq. (1) can be written in Eq. (2).

$$\mathcal{Y} \in \mathbb{R}^{H'' \times W'' \times D''} \tag{2}$$

The symbols of H'', W'', and D'' represent the height, width, and number of kernels of the image convolution results, respectively. If we add bias ($b_{d''}$) values, then Eq. (1) will change to the following Eq. (3).

$$\mathcal{Y}_{i'' \, j'' \, d''} = b_{d''} + \sum_{i'=1}^{H'} \sum_{j'=1}^{W'} \sum_{d'=1}^{D} \mathcal{F}_{i^i \, j^i \, d} \times \mathcal{X}_{i''+i''-1, j''+j'-1, d', d''} \tag{3}$$

If an image input is added by zero elements (padding) on the top (P_h^-), bottom (P_h^+), left (P_w^-), and the right (P_w^+), then image output can be expressed as the following Eqs. (4) and (5).

$$\mathcal{Y}_{i'' \, j'' \, d''} = \sum_{i'=1}^{H'} \sum_{j'=1}^{W'} \sum_{d'=1}^{D} \mathcal{F}_{i^i \, j^i \, d} \times \mathcal{X}_{S_h(i''-1)+i'-P_h^-, S_w(j''-1)+j'-P_w^-, d', d''} \tag{4}$$

$$\mathcal{Y}_{i'' \, j'' \, d''} = b_{d''} + \sum_{i'=1}^{H'} \sum_{j'=1}^{W'} \sum_{d'=1}^{D} \mathcal{F}_{i^i \, j^i \, d} \times \mathcal{X}_{S_h(i''-1)+i'-P_h^-, S_w(j''-1)+j'-P_w^-, d', d''} \tag{5}$$

Based on the Eqs. (1), (3), (4), and (5), the convolution result has size as the following Eqs. (6) and (7).

$$H'' = \left\lfloor \frac{H - H' + P_h^- + P_h^+}{S} \right\rfloor + 1 \tag{6}$$

$$W'' = \left\lfloor \frac{W - H' + P_h^- + P_h^+}{S} \right\rfloor + 1 \tag{7}$$

H' and W' represent kernel row and column sizes, whereas H'' and W'' describe sizes of output convolution using kernel and padding. Furthermore, we conduct a batch normalization of the convolution results followed by the rectified linear unit (ReLu) activation function. We normalize convolutional results by using a batch normalization model, where we subtract the original pixel by using the mean $\mu_{z_{i\,j}}$ and divide it by the standard deviation $\sigma^2_{z_{i\,j}}$ and error tolerance ε. Furthermore, the layer shifts the image input by adding the offset β and scale factor γ as shown in Eq. (8).

$$z''_{i\,j} = \frac{z_{i\,j} - \mu_{z_{i\,j}}}{\sqrt{\sigma^2_{z_{i\,j}} + \varepsilon}} \times \gamma_{i\,j} + \beta_{i\,j} \tag{8}$$

The results of Eq. (8) will be changed the values using a Rectified Linear Unit (ReLU), as shown in Eq. (9).

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{9}$$

Moreover, the Eq. (9) results will be further processed using the pooling model. Pooling has a crucial role in reducing the dimension. There are two pooling models: maximum pooling is known as max pooling, and average pooling Eqs. (10) and (11). We apply all pooling models on the different layer.

$$Y_{i' \, j'' d} = \max_{1 \leq i' \leq H''', 1 \leq j' \leq W'''} x_{i''+i'-1, j''+j'-1, d} \tag{10}$$

And

$$Y_{i' \, j'' d} = \frac{1}{W''' \times H'''} \times \sum_{1 \leq i' \leq H''', 1 \leq j' \leq W'''} x_{i''+i'-1, j''+j'-1, d} \tag{11}$$

Calculating the output when an image passes through a pooling (max) layer

$$P_w = \left(\frac{(W'' - W''')}{S} \right) + 1 \tag{12}$$

$$P_h = \left(\frac{(H'' - H''')}{S} \right) + 1 \tag{13}$$

Based on the Eqs. (12) and (13), we can write quickly that the pooling result has (P_w, P_h, D) size.

In the second convolution, we apply $2D''$ kernels, where the kernel has the same size as the first convolution, which is K_w. We have added some kernels twice as many as the first kernel on the first convolution. We have kept the second convolution results the same as the first pooling size. The second convolution of our proposed method can be seen in Fig. 5. We also conducted a normalization process from the convolution results followed by value transformation using the ReLU activation function. The last operation on the second convolution is pooling, where we occupy maximum pooling to reduce the result sizes, as shown in Fig. 5.

The third convolution has a similar structure to the second, only differs in the number of kernels. We have multiplied three times to the first convolution, whereas the second one is multiplied twice from the first one, as shown in Fig. 6. Additionally, the following process after convolution is the same as the first and second.

The last three convolutions also have a similar structure to the first. The difference is the fourth, fifth, and sixth convolutions employ four, five, and six times from the first number of kernels, whereas the other processes, such as batch normalization, ReLU,

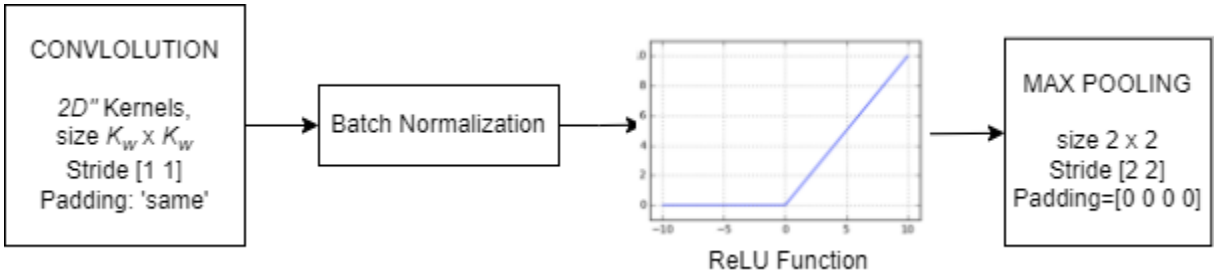

Figure. 5 Our proposed method: the second convolutional, batch normalization, ReLu, and max pooling

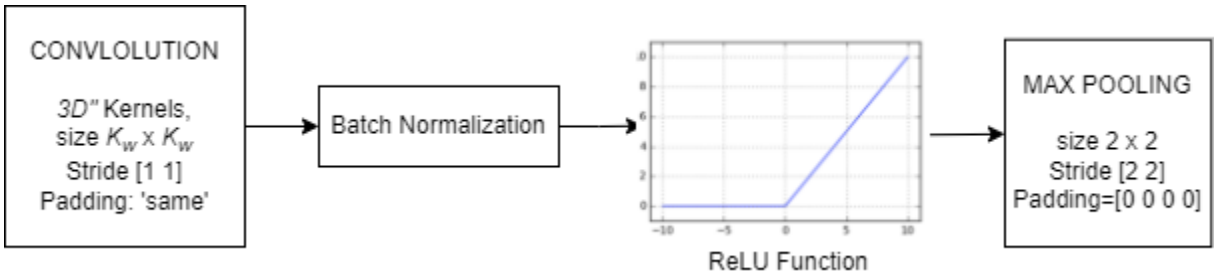

Figure. 6 Our proposed method: the third convolutional, batch normalization, ReLu, and max pooling

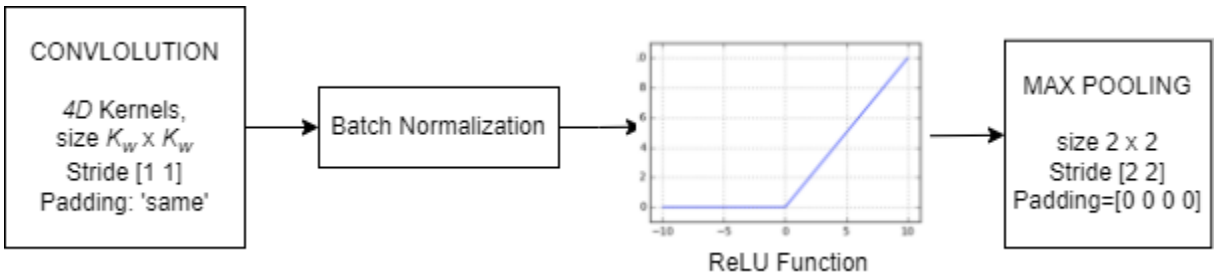

Figure. 7 Our proposed method: the fourth convolutional, batch normalization, ReLu, and max pooling

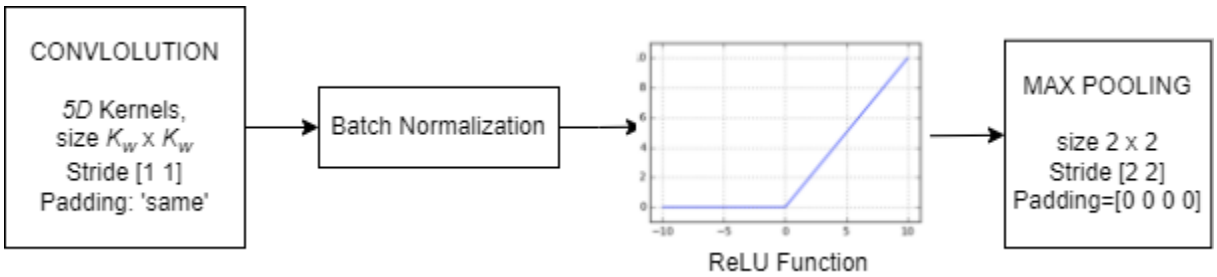

Figure. 8 Our proposed method: the fifth convolutional, batch normalization, ReLu, and max pooling

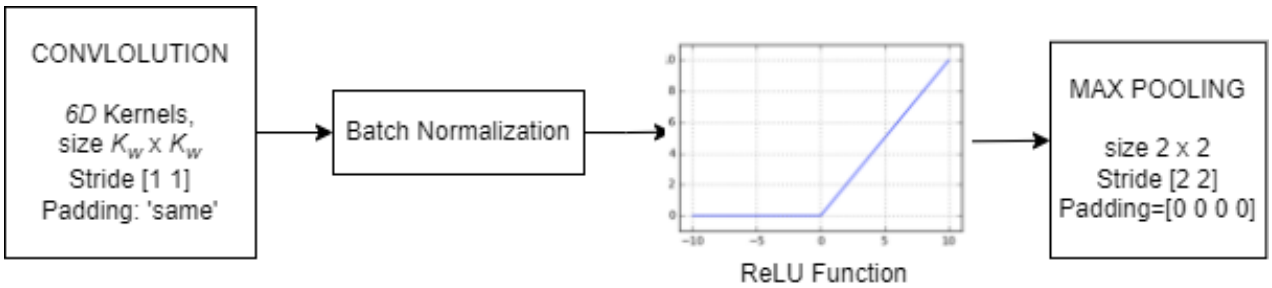

Figure. 9 Our proposed method: the sixth convolutional, batch normalization, ReLu, and max pooling

and max pooling, are the same as the first one, as seen in Figs. 7, 8, and 9.

The last process of the feature extraction layer will be forwarded to the classification layer, which is fully connected, the activation function, and

classification layers. There are two choices in the activation function on the classification layer: Sigmoid and SoftMax. In this paper, we employ the SoftMax activation function as follows.

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \quad (14)$$

2.1 Confusion matrix

A confusion matrix is one of the methods to measure performance. There are four crucial things on the confusion matrix, i.e., true positive, true negative, false positive, and false negative. True positive states that the prediction is positive, and our prophecy is the same as the target (TP). True negative describes that the projection and target are negative (TN). If we predict positive, whereas a target is negative, it is false positive (FP). The last, false negative can appear when we expect a negative, but the reality is positive (FN). Based on four definitions, we can calculate *Accuracy*, *Precision*, and *Recall* as follows

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (15)$$

$$Precision = \frac{TP}{TP+FP} \quad (16)$$

$$Recall = \frac{TP}{TP+FN} \quad (17)$$

Accuracy compares the number of correct prediction results to the target data validation, whereas the comparison between the number of relevant images retrieved and the images present is called *Recall*. In contrast, *Precision* compares the number of relevant images retrieved to the total number of images retrieved by searching.

3. Experimental and discussion

We employ the acute lymphoblastic Leukemia image database (ALL-IDB2), where the datasets consist of two classes: the healthy class has 130 images, and the positive Leukemia class has 130 images. The original image has a size of 257 rows and columns, as shown in Fig. 9. In this experiment, we do not conduct an image segmentation, meaning we processed the original image directly as the input of our proposed architecture. We split the datasets into two parts, firstly: 100 images for each class and the rest as the image testing sets. We use a single GPU, 16 GB RAM, and Core i7 processor to process our proposed architecture.

Table 2. Our experimental scenarios

No	Kernel Size	Number of Kernels	Number of Experiments
1	3 x 3	5, 10, 15, 20, 25, 30	10
2	5 x 5	5, 10, 15, 20, 25, 30	10
3	7 x 7	5, 10, 15, 20, 25, 30	10
4	9 x 9	5, 10, 15, 20, 25, 30	10
5	11 x 11	5, 10, 15, 20, 25, 30	10

We apply five different kernel sizes for each scenario, as seen in Table 2. Firstly, we implement 3x3 kernel sizes, where six convolutions with a varying number of kernels: 5, 10, 15, 20, 25, and 30. Our proposed architecture of CNN is like a pyramid model for both kernel size and the number of kernels. Secondly, we have tried to increase the kernel size, which is 5x5. We also increase the kernel sizes, i.e., the fourth, the fifth, and the sixth scenarios, as seen in Table 2. Based on Table 2, we have conducted fifty experiments, i.e., ten experiments for the first, second, third, fourth, and fifth scenarios.

In the first experiment, as seen in Table 2, we employ a 3x3 kernel size to extract the feature with stride [1 1], where the feature map resulting has the same size as the previous image input. Furthermore, we normalize the feature map before the pooling process. We apply 2x2 max-pooling without padding and [2 2] stride to reduce the feature map. The results show that the proposed architecture has produced at least 96.67% accuracy and 100% accuracy. The average *accuracy* of our proposed architecture is 97.67%. However, our proposed method performs well with a 0.014 standard deviation. Our proposed architecture has slightly different accuracies in the testing data sets. If our testing results have produced a smaller standard deviation, then our proposed architecture has conducted a better performance. Similarly, the *Precision* and *Recall* of our proposed architecture also show a good performance, as shown in Tables 4 and 5, where the average, minimum, and maximum *Precision* and *Recall* are the same as *Accuracy*. In contrast, the differences are in the standard deviation, i.e., 0.0161.

We have also conducted the same experiment (the second until the last) as the first experiment. The difference is kernel size to obtain the feature of the image. The *Accuracy*, *Precision*, and *Recall* can be shown in Tables 3, 4, and 5, respectively. Based on the Tables, we can explain that the maximum (96.67%) and maximum (100%) have the same value, where the differences are the average and standard deviation of the *Accuracy*, *Precision*, and *Recall*. However, the average *Precision* and *Recall* are greater than 97%, as shown in Tables 3, 4, and 5,

Table 3. The Accuracy of our Experimental Results

Exp	Kernel Sizes				
	3 x 3	5 x 5	7 x 7	9 x 9	11 x 11
1st	96.67	98.33	98.33	98.33	98.33
2nd	98.33	96.67	98.33	98.33	98.33
3rd	96.67	98.33	98.33	100.00	100.00
4th	96.67	98.33	100.00	98.33	98.33
5th	100.00	100.00	98.33	98.33	100.00
6th	96.67	100.00	96.67	98.33	100.00
7th	98.33	96.67	100.00	96.67	98.33
8th	96.67	98.33	98.33	100.00	98.33
9th	96.67	98.33	98.33	98.33	100.00
10th	100.00	96.67	98.33	100.00	100.00
Avg	97.67	98.17	98.50	98.67	99.17
Min	96.67	96.67	98.33	96.67	98.33
Max	100.00	100.00	100.00	100.00	100.00
Std	0.0140	0.0123	0.0095	0.0105	0.0088

Table 4. The Precision of our Experimental Results

Exp	Kernel Sizes				
	3 x 3	5 x 5	7 x 7	9 x 9	11 x 11
1st	96.67	98.33	98.33	98.33	98.33
2nd	96.67	96.67	98.33	98.33	98.33
3rd	96.67	98.33	98.33	100.00	100.00
4th	96.67	100.00	100.00	98.33	98.33
5th	100.00	100.00	98.33	98.33	100.00
6th	96.67	100.00	96.67	98.33	100.00
7th	100.00	96.67	100.00	96.67	98.33
8th	96.67	98.33	98.33	100.00	100.00
9th	96.67	98.33	98.33	98.33	100.00
10th	100.00	96.67	96.67	100.00	100.00
Avg	97.67	98.33	98.33	98.67	99.33
Min	96.67	96.67	96.67	96.67	98.33
Max	100.00	100.00	100.00	100.00	100.00
Std	0.0161	0.0136	0.0111	0.0105	0.0086

whereas the standard deviations are about 0.01. It shows that our proposed method has produced consistency in *Accuracy*, *Precision*, and *Recall*, even though we have changed the kernel size. However, the kernel size has influenced the accuracy results. We found 100% accuracy in the fifth scenario five times out of ten experiments, i.e., the third, fifth, sixth, ninth, and tenth.

In contrast, we obtained 100% accuracy in the fourth scenario, i.e., the third, eighth, and last experiments. In the other scenario, we produced 100% accuracy twice, i.e., on the first, second, the third scenario. Based on the experimental results, we can conclude that the greater the kernel sizes used, the higher *Accuracy*, *Precision*, and *Recall* produced. We have calculated trainable parameters for each scenario, as shown in Fig. 6. The first scenario has updated 16.847 trainable parameters, whereas it consists of 15.885 trainable parameters and 962 feature extraction and classification layers. Additionally, in the second scenario, the number of trainable parameters has increased to 44.125. It shows that the different number of parameters is 44.125-16.847=27.278 trainable parameters. A similar condition also occurred in the third, fourth, and last scenarios, where we have updated 87.447, 143.927, and 214.527 trainable parameters, respectively. The greater the kernel size employed for the convolution process, the greater the trainable parameters must be updated. However, the average *Accuracy* also increased according to the increase in kernel size, i.e., 97.67%, 98.17%, 98.50%, 98.67%, and 99.17% for the first, second, third, fourth, and fifth scenarios, respectively. Similarly, it also occurred in *Precision* and *Recall*, as shown in Tables 4 and 5. We have investigated our experimental results; each scenario has produced the standard deviation for *Accuracy*, i.e., 0.0140, 0.0123, 0.0095, 0.0105, and 0.0088 for the first until the fifth scenario. In contrast, *Precision* has produced the standard deviation, i.e., 0.0161, 0.0136, 0.0111, 0.0105, and 0.0086. Similarly, the *Recall* has obtained 0.0161,

Table 5. Recall of our Experimental Results

Exp	Kernel Sizes				
	3 x 3	5 x 5	7 x 7	9 x 9	11 x 11
1st	96.67	98.33	98.33	98.33	98.33
2nd	100.00	96.67	98.33	98.33	98.33
3rd	96.67	98.33	98.33	100.00	100.00
4th	96.67	96.67	100.00	98.33	98.33
5th	100.00	100.00	98.33	98.33	100.00
6th	96.67	100.00	96.67	98.33	100.00
7th	96.67	96.67	100.00	96.67	98.33
8th	96.67	98.33	98.33	100.00	96.67
9th	96.67	98.33	98.33	98.33	100.00
10th	100.00	96.67	100.00	100.00	100.00
Avg	97.67	98.00	98.67	98.67	99.00
Min	96.67	96.67	96.67	96.67	96.67
Max	100.00	100.00	100.00	100.00	100.00
Std	0.0161	0.0131	0.0105	0.0105	0.0117

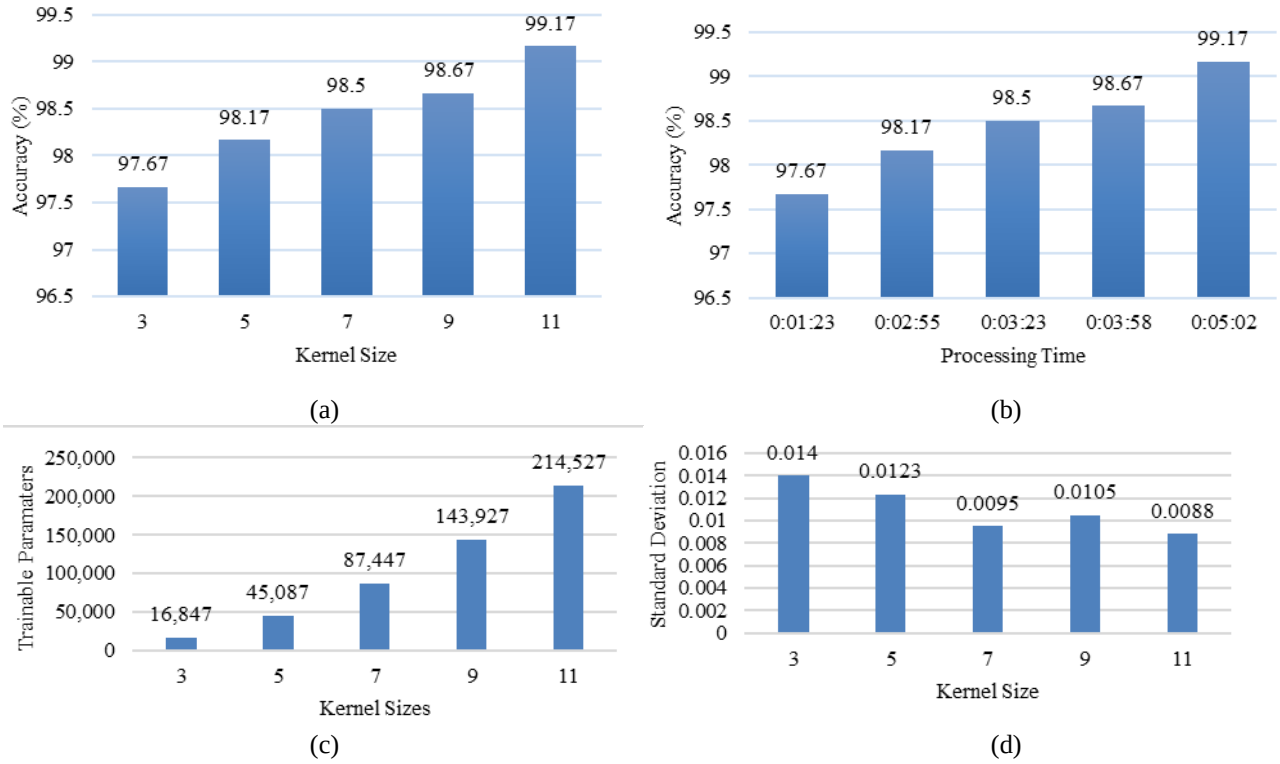

Figure. 10 Correlation accuracy, Kernel size, processing time, standard deviation, and trainable parameters: (a) Correlation accuracy and Kernel size, (b) Correlation accuracy and processing time, (c) Correlation trainable parameters and Kernel sizes, and (d) Correlation standard deviation and Kernel size

0.0131, 0.0105, 0.0105, and 0.0117. The smaller the *Accuracy*, *Precision*, and *Recall* indicate that the results obtained are close to consistency, where the difference in the results is minimal, as seen in the last three Tables 3, 4, and 5.

We have also investigated a correlation between Kernel size, processing time, number of trainable parameters, and standard deviation, as described in Fig. 10.

Fig. 10 demonstrates our experimental results using different Kernel sizes. We have carried out ten experiments for each kernel size, and then our proposed method has recorded the accuracy, standard deviation, processing time, and average accuracy. In this case, we have initialized some parameters, such as Epoch=15; batch size=10; toral Exp=20; learning rate=0.01; val freq=150; and number of training files = 100. Firstly, we have investigated the use of kernel size and average accuracy, as seen in Fig. 10 (a). The results show that using kernel size can impact the accuracy results. The more significant values of the kernel size, the higher average accuracy has been produced by our proposed model. We can demonstrate that the use of 11x11 kernel matrix size has delivered the highest average accuracy (99.17%) than the others, such as 9x9, 7x7, 5x5, and 3x3 kernel sizes. However, the bigger kernel sizes required a longer processing time, as described in Fig. 10 (b). We know that the updated number of trainable parameters influences processing time, and the smaller kernel size is employed, the smaller number of trainable parameters are updated.

On the contrary, the bigger the kernel size, the more trainable parameters are processed, as seen in Fig. 10 (c). We can also see that obtaining the highest average accuracy requires a longer time than the others. We have also demonstrated that using the biggest kernel size has delivered the smallest standard deviation, i.e., 0.0088, as described in Fig. 10 (d).

4. Comparing to other methods

We have recorded the time required for the training process and the number of trainable parameters for each scenario, as seen in Table 6. In the first scenario, we employ a 3x3 kernel size. There are 16.847 trainable parameters on the feature extraction and classification layers that must be updated, where the time required is one minute and twenty-three seconds. The number of trainable is not dependent on the image size. However, it is influenced by the number of kernels and size. In the second scenario, we increase kernel size to 5x5. The kernel size increase has also affected linearly the number of trainable parameters and the time required.

Table 6. Impact of trainable parameters on *accuracy*

Scenario	Kernel Size	Trainable Parameters	Time	*Accuracy*	
				Avg	Max
1st	3x3	16.847	00:01:23	97.67	100
2nd	5x5	45.087	00:02:55	98.17	100
3rd	7x7	87.447	00:03:23	98.50	100
4th	9x9	143.927	00:03:58	98.67	100
5th	11x11	214.527	00:05:02	99.17	100

Table 7. Comparing our proposed method to the others

Method	Accuracy (%)
Shape Features [3]	95.38
Multi distance of GLCM [4]	96.97
CNN and SVM [7]	99.00
Pretrained deep convolutional neural networks [11]	99.50
AlexNet [13]	96.06
Ensemble Network [18]	90.30
Convolutional and Recurrent Neural Network [19]	86.60
Hypercomplex-Valued Convolutional Neural Networks [21]	96.60
Average Accuracy of Our Proposed Method	99.17
Maximum Accuracy of Our Proposed Method	100

Based on Table 6, we demonstrated that the number of trainable parameters is 45.087 and the time required is two minutes and fifty-five seconds, which means that trainable parameters and the time necessary have increased by two times more than before.

Similarly also occurred in the third, fourth, and fifth scenarios, where the increase of the kernel size also affected the trainable parameters and time required. Based on Table 6, we can conclude that the bigger the kernel sizes, the more trainable parameters and the longer time required. The maximum time needed for training our proposed method is only five minutes and two seconds, whereas our proposed architecture only updated 214.527 trainable parameters, as seen in Table 6. However, our proposed method has taken less time than AlexNet [11], where AlexNet needs around thirty-five to train the image data sets. The AlexNet architecture updated 4.097.000 trainable parameters. Based on the above, our proposed method needs less time than AlexNet.

We also compared our *Accuracy* to the others. The lowest average *Accuracy* of our proposed method is 97.67%, whereas the best average *Accuracy* is 99.17%. In this section, we also compare the others, i.e., as seen in Table 7. Several methods have been implemented by other researchers, such as shape features [3], Multi distance of GLCM [4], CNN and SVM [7], Pretrained deep convolutional neural networks [11], AlexNet [13], ensemble network [18], convolutional and recurrent neural network [19], and hypercomplex-valued convolutional neural networks [21].

The results show that our proposed method produced better than the others, except for pre-trained deep convolutional neural networks. However, the maximum accuracy of our proposed method is better than pre-trained deep convolutional neural networks [11].

Our proposed architecture is better than the others because our proposed method has employed a smaller stride size than the kernel matrix size so that lossless information of the object at the image. We have investigated our proposed architecture related to the stride size when the convolution process. We have defined the stride [1 1] for all convolution processes, while the kernel sizes are increased from 3x3, 5x5, 7x7, 9x9, and 11x11. If the stride size is smaller than the kernel size, then the main features of the object can be captured almost perfectly. Therefore, our proposed method can capture wholly of the object. The impact of the small-size stride can increase the processing time required. Somehow, we can reduce the time complexity by decreasing the number of kernels when the convolution processes. Empirically, we can show that the experimental results have produced better accuracy when we apply the bigger kernel size, as seen in Tabel 3, 4, and 5. The accuracies are 97.67%, 98.17%, 98.50%, 98.67%, and 99.17% for the 3x3, 5x5, 7x7, 9x9, and 11x11 respectively. We can see that the classification accuracy increases linearly as the increasing of kernel size. It can be indicated that using the bigger kernel size can perfectly capture an object.

5. Conclusion

Our proposed method produced the minimum average accuracy of 97.67% when we employed a 3x3 kernel size, while the maximum average accuracy was 99.17% when we used an 11x11 kernel size. However, we have produced fife time of 100% accuracy for each kernel size. The experimental results have also demonstrated a correlation between kernel size and accuracy, where the more significant the kernel size used, the better accuracy obtained. The comparing results also show that our proposed method outperformed AlexNet, ResNet, VGG16,

Ensemble Network, Convolutional and Recurrent, ResNet18, Shape Features, Multi distance of GLCM, Hausdrof SVM-based Leukemia Detection, and Hypercomplex-Valued Convolutional Neural Networks.

Conflicts of interest

We would like to declare no conflicts of interest

Author contributions

We have completed our research. The directorate of higher education - ministry of research, technology, and higher education fully funds our research for supporting our research under the fundamental research funding scheme 2022. Finally, our research has been finished by our team. Each person has completed their work as follows: Conceptualization: Arif Muntasa and Rima Tri Wahyuningrum; Methodology: Arif Muntasa and Abdelwahed Motwakel; Literature review: Arif Muntasa, Muhammad Yusuf, and Abdelwahed Motwakel; Image Data preparation: Zabrina Tuzzahra; Architecture model: Arif Muntasa; Algorithm Designer: Arif Muntasa and Wayan Firdaus Mahmudi; Software Implementation: Arif Muntasa and Wayan Firdaus Mahmudi; Experimental designer: Muhammad Yusuf; Validation: Rima Tri Wahyuningrum; Result analysis: Abdelwahed Motwakel, Zabrina Tuzzahra, and Arif Muntasa; Investigation: Arif Muntasa and Muhammad Yusuf; Writing original draft preparation: Arif Muntasa; Writing review and editing: Muhammad Yusuf, Abdelwahed Motwakel, and Arif Muntasa; Supervision: Wayan Firdaus Mahmudi; Project administration: Rima Tri Wahyuningrum.

Acknowledgment

We thank the directorate of higher education – ministry of research, technology, and higher education, for supporting our research under the fundamental research funding scheme 2022. Thank you very much to the Computational Artificial Intelligence Laboratory in the informatics engineering department and the Business Intelligence system laboratory at the Information System Department, Engineering Faculty, University of Trunojoyo Madura, Indonesia.

References

[1] P. S. R. S. M. M. D. Joshi, and P. A. H. Karode, "Detection of Acute Leukemia Using White Blood Cells Segmentation Based on Blood Samples", *(Ijecet), Int. J. Electron. Commun. Eng. Technol.*, Vol. 4, No. 3, pp. 148–153, 2013.

[2] L. Putzu and C. D. Ruberto, "White Blood Cells Identification and Classification from Leukemic Blood Image", in *International Work-Conference on Bioinformatics and Biomedical Engineering*, pp. 18–20, 2013.

[3] A. Muntasa and M. Yusuf, "Color-based hybrid modeling to classify the acute lymphoblastic leukemia", *Int. J. Intell. Eng. Syst.*, Vol. 13, No. 4, pp. 408–422, 2020, doi: 10.22266/ijies2020.0831.36.

[4] A. Muntasa and M. Yusuf, "Multi Distance And Angle Models Of The Gray Level Co-Occurrence Matrix(Glcm) To Extract the Acute Lymphoblastic Leukemia (All) Images", *Int. J. Intell. Eng. Syst.*, Vol. 14, No. 6, pp. 357–368, 2021, doi: 10.22266/ijies2021.1231.32.

[5] D. Kumar, N. Jain, A. Khurana, S. Mittal, S. C. Satapathy, and R. Senkerik, "Automatic Detection of White Blood Cancer From Bone Marrow Microscopic Images Using Convolutional Neural Networks", *IEEE Access*, Vol. 8, pp. 142521-142531, 2020, doi: 10.1109/ACCESS.2020.3012292.

[6] J. E. M. D. Oliveira and D. O. Dantas, "Classification of normal versus leukemic cells with data augmentation and convolutional neural networks", in *VISIGRAPP 2021 - Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2021, pp. 685–692.

[7] L. H. S. Vogado, R. M. S. Veras, F. H. D. Araujo, R. R. V. Silva, and K. R. T. Aires, "Leukemia diagnosis in blood slides using transfer learning in CNNs and SVM for classification", *Eng. Appl. Artif. Intell.*, Vol. 72, pp. 415–422, Jun. 2018.

[8] P. Viswanathan, "Fuzzy C Means Detection of Leukemia Based on Morphological Contour Segmentation", in *Procedia Computer Science*, 2015, pp. 84–90.

[9] J. Laosai and K. Chamnongthai, "Classification of acute leukemia using medical-knowledge-based morphology and CD marker", *Biomed. Signal Process. Control*, Vol. 44, pp. 127–137, Jul. 2018.

[10] S. Mohapatra, S. S. Samanta, D. Patra, and S. Satpathi, "Fuzzy based blood image segmentation for automated leukemia detection", in *2011 International Conference on Devices and Communications, ICDeCom 2011 - Proceedings*, 2011, pp. 1–5.

[11] S. Shafique and S. Tehsin, "Acute lymphoblastic leukemia detection and classification of its subtypes using pretrained deep convolutional

neural networks", *Technol. Cancer Res. Treat.*, Vol. 17, No. 8, pp. 1–7, 2018.

[12] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection", *Prog. Artif. Intell.*, Vol. 9, No. 2, pp. 85–112, 2020.

[13] C. Marzahl, M. Aubreville, J. Voigt, and A. Maier, "Classification of Leukemic B-Lymphoblast Cells from Blood Smear Microscopic Images with an Attention-Based Deep Learning Method and Advanced Augmentation Techniques", in *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*, pp. 13–22, 2019.

[14] B. S. L. J. S. Jung, Wonseok, "Performance Comparison of the Optimizers in a Faster R-CNN Model for Object Detection of Metaphase Chromosomes", *J. Korea Inst. Inf. Commun. Eng.*, Vol. 23, No. 11, pp. 1357–1363, 2019.

[15] A. Honnalgere and G. Nayak, "Classification of Normal Versus Malignant Cells in B-ALL White Blood Cancer Microscopic Images", in *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*, 2019, pp. 1–12.

[16] M. J. Ahmed and P. Nayak, "Detection of Lymphoblastic Leukemia Using VGG19 Model", in *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pp. 716–723, 2021.

[17] Y. Pan, M. Liu, Y. Xia, and D. Shen, "Neighborhood-Correction Algorithm for Classification of Normal and Malignant Cells", in *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*, pp. 73–82, 2019.

[18] F. Xiao, R. Kuang, Z. Ou, and B. Xiong, "DeepMEN: Multi-model Ensemble Network for B-Lymphoblast Cell Classification", in *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*, pp. 83–93, 2019.

[19] S. Shah, W. Nawaz, B. Jalil, and H. A. Khan, "Classification of Normal and Leukemic Blast Cells in B-ALL Cancer Using a Combination of Convolutional and Recurrent Neural Networks", in *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*, pp. 23–31, 2019.

[20] S. Mohapatra and D. Patra, "Automated leukemia detection using hausdorff dimension in blood microscopic images", in *International Conference on* "*Emerging Trends in Robotics and Communication Technologies*", *INTERACT-2010*, pp. 64–68, 2010.

[21] V. Guilherme and M. E. Valle, "Acute Lymphoblastic Leukemia Detection Using Hypercomplex-Valued Convolutional Neural Networks", in *International Joint Conference on Neural Networks (IJCNN 2022)*, pp. 1–8, 2022.